



资深网络技术工程师、全国网管技能水平考试认证专家、国内优秀IT图书作者王达老师最新力作，51CTO技术社区鼎力推荐！

结合最新技术，全面、系统、深入阐述计算机网络的体系结构、功能实现原理和通信协议实现原理

包含近600幅图表、形象的比喻和丰富的案例，使得本书通俗易懂，极大程度地降低了读者的学习成本



提供教学用PPT

深入理解 计算机网络

Understanding Computer Networks

王达 著



机械工业出版社
China Machine Press

深入理解计算机网络

王达 著

ISBN: 978-7-111-41188-8

本书纸版由机械工业出版社于2013年出版，电子版由华章分社（北京华章图文信息有限公司）全球范围内制作与发行。

版权所有，侵权必究

客服热线: + 86-10-68995265

客服信箱: service@bbbvip.com

官方网址: www.hzmedia.com.cn

新浪微博 @研发书局

腾讯微博 @yanfabook

目录

前言

为什么写这本书

读者对象

如何阅读本书

本书的特色

勘误和支持

致谢

第1章 数制与编码

1.1 数制概述

1.2 不同数制间的相互转换

1.3 二进制数运算

1.4 二进制数的表示形式

第2章 计算机网络概述

2.1 计算机网络概述

2.2 计算机网络的分类

2.3 计算机网络拓扑结构

第3章 计算机网络体系结构

3.1 典型计算机网络体系结构

3.2 计算机网络体系结构通信原理

3.3 网络体系结构的设计考虑

3.4 网络体系结构中的通信协议

第4章 物理层

4.1 物理层概述

4.2 数据通信基础

4.3 数据传输速率与信道带宽

4.4 数字基带信号编码

4.5 信号调制与解调

4.6 物理层传输介质

4.7 信道多路复用技术

4.8 物理层接口

第5章 数据链路层

5.1 数据链路层基础

5.2 数据链路层主要功能及实现原理

5.3 差错控制方案

5.4 流量控制

5.5 面向字符的BSC协议

5.6 面向比特的SDLC和HDLC协议

5.7 面向字符的PPP同步传输协议

5.8 数据链路层主要网络设备

第6章 介质访问控制子层

6.1 MAC子层基础

6.2 CSMA介质访问控制原理

6.3 CSMA/CD介质访问控制原理

6.4 局域网标准及以太网帧格式

6.5 标准以太网规范及体系结构

6.6 快速以太网规范及体系结构

6.7 千兆以太网规范及体系结构

6.8 万兆以太网规范及体系结构

6.9 IEEE 802.1d协议

6.10 IEEE 802.1q协议

6.11 IEEE 802.1w协议

6.12 IEEE 802.1s协议

6.13 IEEE 802.1x协议

6.14 主要WLAN标准与技术

第7章 网络层

7.1 网络层概述

7.2 网络层数据交换及相关技术

7.3 网络层协议及报文格式

7.4 路由和路由算法

7.5 几种主要的路由算法解析

7.6 网络拥塞控制方法和原理

7.7 网络层设备及主要技术

第8章 IP地址和子网

8.1 IPv4地址

8.2 IPv4子网划分与聚合

8.3 IPv4 NAT基础

8.4 IPv6地址基础

8.5 IPv6地址类型

8.6 IPv6地址自动配置

第9章 路由协议及工作原理

9.1 RIP路由协议

9.2 OSPF路由协议

9.3 IS-IS路由协议

9.4 BGP

第10章 传输层

10.1 传输层概述

10.2 传输层服务功能

10.3 TCP概述

10.4 TCP的可靠传输

10.5 TCP的流量控制

10.6 TCP的拥塞控制

10.7 UDP概述

第11章 应用层

11.1 应用层概述

11.2 Web服务基础

11.3 DNS服务

11.4 DHCP服务

11.5 电子邮件服务

前言

本书是笔者从业二十余年、从事图书创作十余年的工作经验和技术积累的结晶，是对十余年来一直默默支持我的全国百万读者的真诚回馈。同时，本书也是笔者这十多年来付诸心血最多（整整一年专职创作时间）、寄予希望最大的一部重头之作，期待能为国家的计算机网络专业教育尽一份绵薄之力。

为什么写这本书

其实很久以前就有了写这本书的动机了，但由于我深知写作这本书的难度很大，再加上自己还在写作其他图书，写作任务一直非常繁重，所以就耽误了。不过，或许今天写这本书正是时候，一则笔者又多经过了几年的技术学习和工作经验的积累，书稿的质量可能比以前更高；二则目前计算机网络专业越来越边缘化了，已成为了所有IT人员的必修课，所以现在对要求有一本高质量、通俗易懂的专注于计算机网络原理和基础知识的教材的呼声比以前更高了。综合起来就是以下三点：

1.学子的呼唤：“零基础”不应只是一句宣传口号

计算机网络原理和基础知识类的课程一直是广大计算机网络专业的读者最头痛的一门课程。因为这类课程不仅相对枯燥乏味，而且教材中的技术原理解释普遍晦涩难懂。也正因如此，现在许多计算机网络专业的大学生，毕业后仍对这方面的知识一知半解，走上工作岗位后遇到实际的网络问题很难从原理方面分析出故障原因，更别说排除网络故障了。

虽然国内这方面的教材非常多，也不乏一些经典著作，但经笔者分析后认为大部分存在这样或那样的不足，要么通俗性较差，要么内容上过于浅显，更多的是照搬理论，难以使网络专业学生比较轻松地掌握全面、系统、专业的计算机网络原理和基础知识。但作为一名老的网络职业工作者和有着十几年计算机网络专业图书创作经验的老作者，深知这样一本看似非常基础，甚至有一些人认为非常简单的教材，要真正写出水平、写出权威并非易事，特别是在通俗性方面。现在许多书都把“零基础”当作卖点在宣传，但真正能做到零基础，并且在内容上有一定深度的书却并不多见。

另外，以前学计算机网络原理和网络基础知识的人可能大多数是计算机网络专业的学生，但随着计算机网络应用的普及，计算机网络知识几乎已成为所有IT专业必修的基础课程。而那些非计算机网络专业的学生对计算机网络可以说是真正的零基础，所以对这类教材在通俗性方面的要求会更高。要把那么深奥的计算机网络原理讲得能让这

些零基础的读者理解和接受，难度就更是难以想象的了，这点笔者在创作过程中深有体会。尽管笔者在这方面也没有过深的造诣，但本着对信任、支持笔者的百万读者负责，怀抱着百万读者的期待和笔者自己二十多年的学习和工作经验积累，花了整整一年全职的创作时间写下了这本笔者认为在某些方面，特别是在通俗性方面有所提高的著作，力争使零基础的网络“菜鸟”也能轻松掌握复杂、深奥的计算机网络原理。希望这本书不会令广大读者朋友失望。

2.时代的变迁：不懂计算机网络，你不敢说自己是ITer

如果十年前你听到同行们都在说“不懂计算机，都不敢说自己是ITer”，那么十年后的今天，你所听到的一定就是“不懂计算机网络，都不好意思说自己是ITer”。更有人甚至会说“不懂计算机网络，就是现代文盲”。这些观点虽然可能有些偏颇，但也足以说明在全面信息化的今天，计算机网络在整个IT行业的重要性和基础性，它不再仅是网络专业人士必须掌握的，所有ITer，甚至所有现代人都应该掌握。

以上虽然看似口号，但却实实在在地反映了当前整个IT行业都是以计算机网络作为中心和基础平台的这样一个现状。十年前，几乎所有的IT开发和应用都是以单一的计算机系统为平台的，几乎所有的计算机程序的运行环境都是单台计算机。十年后的今天，随着互联网接入的普及和宽带接入速度的提高，以及互联网和企业网络技术在应用上的普及与发展，一切都发生了变化。

过去单一的计算机系统平台根本无法满足当前无处不在、各种各样的网络应用需求，绝大多数IT开发和应用平台都转向了计算机网络这个无边的大平台。现在个人和企事业单位所进行的各项IT应用绝大多数都是基于计算机网络的，如浏览网页、收发邮件、写博客、写微博、网上购物、网上看电影/电视、网上玩游戏、网上听音乐、网络电子商务、网络营销、企业网络远程互联、网络会议、网络直播等。似乎我们现在所做的一切一切都离不开计算机网络，计算机网络成了实实在在的IT计算中心和基础应用平台。

现在基于单一计算机系统的应用已非常少了，且随着云计算、物联网这样的新型网络技术的应用和普及，可以十分清楚地预见，计算机网络这个平台才是整个IT行业发展的根本。就连现在我们仍然基于单机操作的办公应用软件，在不久的将来都可能全由云计算服务提供商通过互联网集中提供，再加上迅猛发展的移动互联网，计算机网络的基础地位将得到进一步巩固。到那时，如果连何为计算机网络都不懂，简单的计算机网络故障排除还要求助于人，这不就是现代文盲吗？你还敢说你是ITer吗？

3.职业的挑战：计算机网络基础原理，网络职业发展的真正瓶颈

形势摆在我们所有ITer面前，但国内的现状却不怎么令人满意。先且不说所有IT行业，就是专门从事网络管理，或者网络工程行业的网络管理员和网络工程师，在计算机基础原理方面能比较深入地说出

个一、二、三来的也没多少，碰到一个网络故障能从原理上进行全面分析的人更是少之又少，至少我所了解的是这样。可能有些人会说，会配置和管理网络不就行了？他们认为那些深奥的基础理论没什么用。其实说这样话的人还是不是很懂得网络管理和网络工程的真正职责，不是很理解这些网络基础原理的本质和重要性。网络管理的主要职责就是维护，在出现了网络故障时能快速、准确地排除故障，网络工程的主要职责就是为用户设计一个实用、符合各项标准，且稳定的系统。

很难想象，一个网络基础理论不扎实的网络管理人员如何能快速、准确地进行网络故障分析和排除，网络工程人员又如何能设计出一个符合标准、符合用户应用需求且能长时间保持稳定的系统。对于从事各种网络应用程序开发的程序员们来说，网络基础理论同样非常重要，一个不懂得网络体系结构，以及各层功能实现原理和应用接口的程序员，怎么可能设计出一个符合对应网络应用标准的应用程序？又怎么可能被用户认同？

随着计算机网络应用的不断高速发展，随着一大批快速成长型中小企业的高速发展，相信在不久的将来，全国将有无数企事业单位急需高水平、全面掌握基于计算机网络基础平台的IT设计和IT管理专业人才，到那时必将是一场残酷的职业竞争。如果连计算机网络基础原理都没有比较好的掌握，在起跑线上就输了，还如何参与竞争？

基于以上分析，我们可以十分清楚地知道，要成为一名合格的IT专业人才，无论你是从事IT应用开发，还是从事网络管理和网络工程设计，计算机网络基础原理都将是你的必修课！不要让自己输在起跑线上！

读者对象

本书内容看似非常专业、深奥，但这方面的知识现在已成为所有ITer的必修课。本书适合以下读者阅读，每类读者都可以通过阅读本书获得相应的收益：

□所有想从事网络管理、网络工程设计的准网络管理员、准网络工程师

□所有在网络职业发展道路遇到基础理论瓶颈的在职网络专业人士

□所有大中专院校的IT专业学生

□所有想学习计算机网络技术的“菜鸟”

如何阅读本书

本书虽然在知识讲解上已力争尽可能通俗化，但里面的知识点毕竟相当专业，所以在阅读本书时，建议注意以下几个方面：

□不要刻意追求阅读速度

书中有的专业技术原理还是比较复杂的，建议大家在阅读时要一章、一节地消化，不要刻意追求阅读速度。一定要静下心来，认真阅读，千万别一目十行。

□结合书中的示例阅读

本书所介绍的每项技术原理都会结合一些类比、演示示例，阅读者一定要仔细阅读示例讲解的每一步，最好自己也跟着示例进行计算、分析，以加深对原理的理解。

□坚持，坚持，再坚持

尽管书中在通俗化方面已有较好的体现，但书中的内容毕竟全是基础理论，仍不能完全克服枯燥性，远不如图形操作界面那么简单明了，所以在阅读本书时一定要坚持，要静下心来学习，千万别半途而废。

□选学内容可先不学

本书第1章和第9章属选学内容，主要是为已有一些基础的读者而准备的，所以如果你基础不是太好，可先跳过这两章，等以后有兴趣时再来学习。

本书的特色

“要登堂入室，必须先打开一扇门，或者一扇窗”，本书就是你登入计算机网络神圣殿堂的那扇门或窗。与同类图书对比，本书具有以下特色：

□通俗易懂

这是本书最大的特色。为了能把复杂的技术原理讲得通俗易懂，书中不仅使用了大量的现实生活事例作为说明性的比喻，还列举了许多实例。同时，本书近600幅插图、近100个表格，可以帮助读者朋友更加直观地分析和理解各种复杂的技术原理。这是国内其他同类图书所没有的。

□全面系统

本书的内容应该是同类图书中内容最全面、最系统的，不仅讲了目前主流的TCP/IP体系结构中的相关技术，还同时兼顾了OSI/RM和局域网体系结构的相关技术。更重要的是还包括了与各层对应的一些主要网络基础知识。真正做到“一本在手，网络无忧”。

□专业深入

本书对所写到的每一部分内容都从专业角度进行了非常深入的剖析，使读者朋友不会有在阅读其他同类图书时所有的许多重要知识点都“一笔带过”的感觉。笔者在写作之初就确立了“绝不一笔带过那些重要的知识点”的目标。

□条理清楚

本书无论是从各章节内容安排上，还是从各小节内容组织上，条理都是比较清楚的。在本书中，对于一些比较复杂的内容都分出了多个小标题，重点突出，这样可以使读者朋友更加清楚地理解所介绍的内容，不会有整页或者几页都找不到主题、抓不住重点的现象。

勘误和支持

本书由王达主笔并统稿，参加编写、校验和排版的人员还有：何艳辉、王珂、沈芝兰、马平、何江林、刘凤竹、卢京华、周志雄、洪武、高平复、周建辉、孔平、尚宝宏、姚学军、张磊、刘学、李翔、王娇、李敏、吴鹏飞、宋希岭、刘中洲、潘朝阳、刘伟、曾平辉、李京杨、张跃、周平辉、王新宇、王薄、韩大为、宋宝强、史鹏宇、陆伟等。笔者在此对以上各位老师一并表示最由衷的谢意！尽管我们花了大量时间和精力校验，但由于水平有限，书中难免存在一些错误和瑕疵，敬请各位批评指正，万分感谢！

另外本书读者可以通过以下渠道享受相关服务：

□多个专家博客和认证微博

笔者的主要博客：<http://winda.blog.51cto.com>、
http://blog.csdn.net/lycb_gz、
<http://blog.chinaunix.net/uid/10659021.html>。每个博客里面都有数百篇各方面的专业技术和职业指导文章，以及大量我以前所出版的图书的精彩试读文章。读者朋友不仅可以在里面学习各方面的知识，还可以直接向笔者提问。

笔者的两个微博：weibo.com/winda（新浪微博）、
t.qq.com/winda2010（腾讯微博）。

□超级QQ读者群

为方便全国各地读者交流，专门为本书读者新建了一个超大型、可容纳2000人的超级QQ读者群：196652938。由于读者众多，请尽快购买、加入，否则可能很快就没有位子了（加入时请注明本书名称）。

□授课PPT免费下载

为了支持高校和培训机构老师讲课，本书为各位老师提供了授课PPT，需要的朋友可以在机械工业出版社华章公司官网（www.hzbook.com）上下载，也可直接与笔联系获取（QQ：93220994，邮箱：lycb_gz@vip.sina.com）。

致谢

本书是笔者与机械工业出版社合作出版的第一部图书，感谢机械工业出版社华章公司，以及杨福川老师给予我的这次十分难得的合作机会。由于本书内容较多，专业性较高，出版时间又非常紧，所以特别要感谢杨福川老师专门为本书抽调的精干编辑力量，及他在本书上线前做的大量推广工作；感谢孙海亮等其他所有编辑老师对本书的辛勤付出，我经常发现他们加班加点在编辑这部图书。期待本书能取得好的成绩，也期待通过此部图书合作的成功，为笔者与机械工业出版社展开更广泛的合作打下坚实基础。

王达

第1章 数制与编码

本章可作为选学内容，所介绍的知识仅是为了帮助大家理解本书以后章节中涉及的二进制、八进制、十六进制的内容，特别是各种信息编码、IP地址格式转换、MAC地址格式转换等内容。本章的知识对于数据包分析非常有用。如果你对这些内容已掌握了，可直接跳过本章。

“数制”是“数据进制”的简称，也就是表示数据逢几进位的意思，如我们常用的十进制就是逢十进位。当然，数制的类型远不只十进制，在计算机系统中常见的还有二进制、八进制和十六进制这三种。与数制关系最密切当然就是数据编码了。数据编码主要包括原码、反码和补码三种，它们用于以不同形式表示数据，当然这主要是一些特殊的应用需求，如在进行校验和（checksum）计算时，就需要用于原码计算。反码和补码是为了在计算机中表示负数才出现的。本章的主要内容就是数制与编码的概念，以及二进制的运算和表示形式。

1.1 数制概述

“数制”就是“数据进制”的简称，是指数据的进位计数规则，又称“进位计数制”，简称“进制”。本节先来简单地了解一些常见的数制类型及其特点。

1.1.1 常见数制类型及表示方法

日常生活中我们经常使用的数是十进制的，如我们拿的3000元工资，市场1.5元/斤的菜价等。之所以称其为十进制，是因为这类数是逢十进一的。除了十进制计数以外，还有许多其他进制的计数方法。在计算机中常见的还有二进制、八进制、十六进制等制式。这三种进制的数在进行加法运算中分别是逢二、八、十六进一，这就是前面所说的进位计数规则。关于如何理解这些不同数制类型数据的加法运算，在本章后面将有专门介绍。

其实数制类型远不止这么几种，如我们以60分钟为1小时，60秒为一分钟，用的就是六十进制计数法；一天之中有24小时，用的是二十四进制计数法；而一星期有7天，用的是七进制计数法。

虽然数制类型可以有很多种，但在计算机通信中通常遇到的仍是以上提到的二进制、八进制、十进制和十六进制这四种。在一种数制中所能使用的数码的个数称为该数制的“基数”，也就是对应数制类型的名称，如二进制的基数为“2”，八进制的基数为“8”，十进制的基数为“10”，十六进制的基数也就是“16”。这里所说的“基数”其实就是前面所说的进位计算规则，如我们常见的十进制数是逢十进一，二进制数是逢二进一，.....

既然有不同的数制，那么在计算机程序中给出一个数时就必须指明它属于哪一种数制，否则计算机程序就不知道该把它看成哪种数了。如12300这个数，既可能是十进制、又可能是八进制或者十六进制，所以“数”需要有专门的标志来进行区别。下面分别予以介绍。

(1) 十进制 (Decimal)

十进制是日常生活中常用的数制类型，基数是10，也就是它有10个数字符号，即0、1、2、3、4、5、6、7、8、9。其中最大数码是“基数”减1，即 $10-1=9$ ，最小数码是0。十进制数的标志为D，如(1250)D，也可用下标“10”来表示，如 $(1250)_{10}$ （注意是下标）。

(2) 二进制 (Binary)

二进制是计算机运算时所采用的数制，基数是2，也就是说它只有两个数字符号，即0和1。如果在给定的数中，除0和1外还有其他数（例如1061），那它就绝不会是一个二进制数了。二进制数的最大数码也是基数减1，即 $2-1=1$ ，最小数码也是0。二进制数的标志为B，如(1001010)B，也可用下标“2”来表示，如 $(1001010)_2$ （注意是下标）。

(3) 八进制 (Octal)

八进制的基数是8，也就是说它有8个数字符号，即0、1、2、3、4、5、6、7。对比十进制可以看出，它比十进制少了两个数“8”和“9”，这样当一个数中出现“8”和（或）“9”时（如23459），那它也就绝不是八进制数了。八进制数的最大数码也是基数减1，即 $8-1=7$ ，最小数码也是0。八进制数的标志为O或Q（注意它特别一些，可以有两种标志），如（4603）O（注意是字母O，不是数字0）、（4603）Q，也可用下标“8”来表示，如（4603）₈（注意是下标）。在C、C++这类语言中规定，一个数如果要指明它采用八进制，必须在它前面加上一个0，如：123是十进制数，但0123则表示采用的是八进制。

（4）十六进制（Hexadecimal）

十六进制数用得比较少，最新的IPv6地址就是采用16进制来表示的（IPv4地址通常采用十进制表示）。在注册表中也会用到16进制，所以了解十六进制还是非常重要的。

十六进制的基数是16，也就是说它有16个数字符号，除了十进制中的10个数外，还使用了6个英文字母，这16个数字和字母依次是0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F（与大小写无关）。其中A至F分别代表十进制数的10至15。如果数据中出现了字母之类的符号，如63AB，则一定不会是八进制或十进制，而是十六进制了。它的最大的数码也是“基数”减1，即 $16-1=15$ （为F），最小数码也

是0。十六进制数的标志为H，如(4603) H，也可用下标“16”来表示，如(4603)₁₆（注意是下标）。十六进制数也常常用前缀0x来表示（注意是数字0，而不是字母O）。在C、C++这类编程语言中也规定，16进制数必须以0x开头。比如0x10表示一个十六进制数，而不是八进制或者十进制的10。

经验之谈 既然在计算机中使用的是二进制，那为什么还要十进制、八进制和十六进制呢？其实这都不是计算机自身要求的，因为在计算机运算中使用的全都是二进制。之所以还需要这些数制，完全出于表达和识别的方便性考虑。因为大多数的数据用二进制表示太长了，如一个C、C++等编程语言中的int（整数）类型的数据要占用4个字节，也就是32位。比如100，用int类型的二进制数表达将是：0000 0000 0000 0110 0100。这还是一个比较小的数，如果数更大，则会更复杂。试想一下，要写这么长，估计没几个人会喜欢，于是就有了可以更简便表示的十进制、八进制和十六进制了。所以，像C、C++这类语言没有提供在代码中直接写二进制数的方法，而是普遍采用八进制或十六进制。

那为什么不是其他进制类型，如九或二十进制呢？原因就在于2、8、16，分别是2的1次方、3次方、4次方，这就使得这三种进制之间可以非常直接地互相转换。八进制或十六进制缩短了二进制数，但保持

了二进制数的表达特点。在下面关于进制转换的介绍中，你可以发现这一点。

1.1.2 不同数制之间的对应关系

表1-1所示是以上介绍的二进制、十进制、八进制和十六进制这四种常在计算机中使用的数制的对应关系。注意，八进制没有8和9，二进制1000对应八进制的10，而不是想象中的8，二进制1001对应的八进制数是11，而不是想象中的9，这就是进位造成的。这个表很重要，大家最好全部记下来，特别是这几种数制的对应关系。

表 1-1 不同数制的对应关系

二进制数	对应的十进制数	对应的八进制数	对应的十六进制数
0	0	0	0
1	1	1	1
10	2	2	2
11	3	3	3
100	4	4	4
101	5	5	5
110	6	6	6
111	7	7	7
1000	8	10	8
1001	9	11	9
1010	10	12	A
1011	11	13	B
1100	12	14	C
1101	13	15	D
1110	14	16	E
1111	15	17	F

1.2 不同数制间的相互转换

同一个数在一些环境中（如我们要进行子网划分，或者计算子网掩码时，或者对信息进行编码时等）可能要用不同数据制形式来表示，这就涉及数制间的转换问题了。下面是常见的十进制、二进制、八进制、十六进制之间的转换方法。

1.2.1 非十进制数转换成十进制数

非十进制数转换成十进制数就是把这些非十进制数按位以对应的权值（注意：要区分整数位和小数位）展开，然后相加即得出相应的十进制值。本节后面介绍的各种非十进制转换成十进制的方法都是按照这种方法进行的。

“权值”是指对应数值位的进制幂次方数，如二进制整数中第0位（最低位，也就是整数最右边的那位）的权值是2的0次方，第1位的权值是2的1次方……同理在八进制整数中第0位的权值是8的0次方，第1位的权值是8的1次方……，依此类推。但每位的权值会因是整数位还是小数位而不同：

□整数的第0位（也就是最低位）的权值为对应进制的0次方，最高位的权值为对应进制的 $n-1$ 次方。

□小数的第一位（最靠近小数点的那位，也是小数的最高位）的权值为对应进制的-1次方，最后一位（最右边的那位，也即小数的最低位）的权值为对应进制的-n次方。

1.二进制转换为十进制

二进制转换成十进制的方法，大家可能早就有所了解了，如在IPv4地址计算时就经常进行这样的操作。转换的方法比较简单，只需按它的权值展开即可。展开的方式是把二进制数首先写成加权系数展开格式，然后按十进制加法规则求和。这种方法称为“按权相加”法。

二进制整数部分的一般表现形式为： $b_{n-1} \dots b_1 b_0$ （共n位），按权相加展开后的格式为（注意，展开式中从左往右各项的幂次是从高到低下降的，最高位的幂为n-1，最低的幂为0）：

$$b_{n-1} \times 2^{n-1} + b_{n-2} \times 2^{n-2} \dots + b_1 \times 2^1 + b_0 \times 2^0$$

如二进制数 $(11010)_2$ 的按权相加展开格式为：

$$1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 16 + 8 + 0 + 2 + 0 = (26)_{10}$$

二进制小数部分的幂次是反序排列的（也就是与整数部分的幂次序列相反，从左往右其绝对值是从低到高上升的），且为负值，最高位幂次（也就是最靠近小数点的第一个小数位的幂次）为“-1”。如二

进制小数部分的格式为： $0.b_{n-1} \dots b_1 b_0$ ，则按权相加后的展开格式为：

$$b_{n-1} \times 2^{-1} + b_{n-1} \times 2^{-2} \dots + b_1 \times 2^{-(n-1)} + b_0 \times 2^{-n}$$

如 $(0.1011)_2$ 的按权相加展开格式为：

$$1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4} = 0.5 + 0 + 0.125 + 0.0625 = (0.6875)_{10}$$

2.八进制转换为十进制

八进制转换成十进制也是采取“按权相加”法，只是这里的权值是8的相应幂次方。如八进制整数部分的格式为： $b_{n-1} \dots b_1 b_0$ ，则按权值相加，展开后的格式就为（从左往右幂次是从高到低下降的）：

$$b_{n-1} \times 8^{n-1} + b_{n-2} \times 8^{n-2} \dots + b_1 \times 8^1 + b_0 \times 8^0$$

如八进制数 $(26356)_8$ 的按权值相加展开格式为：

$$2 \times 8^4 + 6 \times 8^3 + 3 \times 8^2 + 5 \times 8^1 + 6 \times 8^0 = 8192 + 3072 + 192 + 40 + 6 = (11502)_{10}$$

八进制小数部分的幂次也是反序排列的（也就是与整数部分的幂次序列相反，从左往右其绝对值是从低到高上升的），且为负值，最低幂次（也就是最靠近小数点的第一个小数位的幂次）为“-1”。如八进制小数部分的格式为： $0.b_{n-1} \dots b_1 b_0$ ，则按权相加后的展开格式为：

$$b_{n-1} \times 8^{-1} + b_{n-2} \times 8^{-2} \dots + b_1 \times 8^{-(n-1)} + b_0 \times 8^{-n}$$

如 $(0.257)_8$ 按权相加的展开格式为:

$$2 \times 8^{-1} + 5 \times 8^{-2} + 7 \times 8^{-3} = 0.25 + 0.078125 + 0.013671875 = (0.341796875)_{10}$$

3.十六进制转换为十进制

十六进制转换成十进制的方法也是采取“按权相加”法，只是这里的权值是16的相应幂次方。如十六进制整数部分的格式为： $b_{n-1} b_{n-2} \dots b_1 b_0$ ，则按权相加展开后的格式就为（从左往右幂次是从高到低下降的）：

$$b_{n-1} 16^{n-1} + b_{n-2} \times 16^{n-2} \dots + b_1 \times 16^1 + b_0 \times 16^0$$

如十六进制数 $(26345)_{16}$ 的按权相加，展开后的格式为:

$$2 \times 16^4 + 6 \times 16^3 + 3 \times 16^2 + 4 \times 16^1 + 5 \times 16^0 = 131072 + 24576 + 768 + 64 + 5 = (156485)_{10}$$

十六进制小数部分的幂次也是反序排列的（也就是与整数部分的幂次序列相反，从左往右其绝对值是从低到高上升的），且为负值，最低幂次（也就是最靠近小数点的第一个小数位的幂次）为“-1”。如十六进制小数部分的格式为： $0.b_{n-1} \dots b_1 b_0$ ，则按权相加后的展开格式为:

$$b_{n-1} \times 16^{-1} + b_{n-2} \times 16^{-2} \dots + b_1 \times 16^{-(n-1)} + b_0 \times 16^{-n}$$

如 $(0.25A)_{16}$ 按权值相加，展开后的格式为：

$$2 \times 16^{-1} + 5 \times 16^{-2} + 10 \times 16^{-3} = 0.125 + 0.0234375 + 0.00244140625 =$$

$$(0.15087890625)_{10}$$

4.同步练习

- 1) 把 $(01110100)_B$ 、 $(11101001000.10111)_B$ 转换成十进制；
- 2) 把 $(1076)_O$ 、 $(6374.65)_Q$ 转换成十进制；
- 3) 把 $0x7A8C$ 、 $0x259B.25$ 转换成十进制。

1.2.2 十进制数转换成非十进制数

十进制数转换成非十进制数的方法是：整数部分的转换用“除基取余法”，也就是用基数相除，然后反序（由后向前取）取余数；小数部分的转换用“乘基取整法”，也就是用基数相乘，然后正序（由前向后取）取整数。这里的“基数”就是对应的数制，如二进制的基数为2，八进制的基数为8，十六进制的基数为16。

1.十进制转换为二进制

这里分别对十进制整数和十进制小数转换成二进制进行介绍。

1) 十进制整数转换成二进制的方法

十进制整数转换为二进制的方法是：采用“除2逆序取余”法（采用短除法进行）。也就是先将十进制数除以2，得到一个商数（也是下一步的被除数）和余数；然后再将商数除以2，又得到一个商数和余数；以此类推，直到商数为小于2的数为止。然后从最后一步得到的小于2的商数开始将其他各步所得的余数（也都是小于2的0或1）排列起来（俗称“逆序排列”）就得到了对应的二进制数。

注意 这里与下面的小数转换有些不一样，这里要包括最后得到的小于2的商数，而小数转换中是不需要包括最后的积的，只包括各步

得到的整数部分，后面的十进制整数转换为八进制、十六进制也一样。

图1-1所示为十进制整数48转换成二进制数时依次除2的过程，在每步的最右边显示的是各步商数除2所得到的余数，最后一步的商数为1，因为它小于2，所以不能再除了。然后从最后得到商数（1）开始依次向上把其他各步除2得到的余数排列起来，就得到最后48转换成二进制时的结果为 $(110000)_2$ 。同理，图1-2所示的十进制数250转换成二进制数后的结果就为 $(11111010)_2$ 。

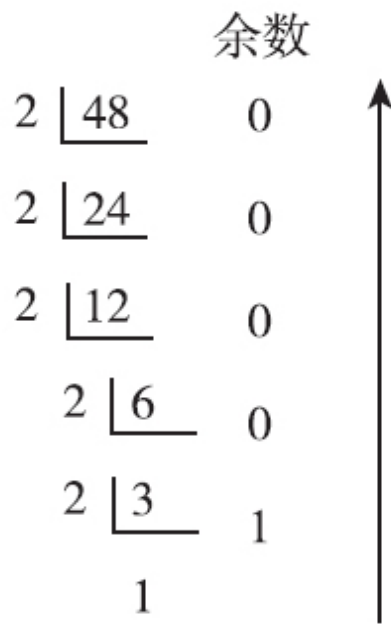


图 1-1 十进制整数48转换成二进制整数的步骤

	余数
2 250	0
2 125	1
2 62	0
2 31	1
2 15	1
2 7	1
2 3	1
1	




图 1-2 十进制整数250转换成二进制整数的步骤

2) 十进制小数转换成二进制的方法

十进制小数转换为二进制的方法是采用“乘2正序取整”法。也就是用2乘十进制小数，得到一个积，然后将积的整数部分取出作为相应步骤得到的整数；再用2乘余下的小数部分，又得到一个积；然后再将这个积的整数部分取出；以此类推，直到积中的小数部分为零，或者达到所要求的精度为止；最后把各步取出的整数部分（仅需要各步得到的整数部分，不需要最后没有取整的小数部分）按正序排列起来，即先取的整数作为二进制小数的高位，后取的整数作为低位。

图1-3的左、上图是分别将十进制小数0.125和0.625转换成二进制的过程，最后得到的二进制数就是从最开始得到的整数值开始，一直到最后得到的整数值（也就是自上而下的顺序，与整数转换中取余的顺序相反）。0.125和0.625最后的二进制值分别为 $(0.001)_2$ 和 $(0.101)_2$ （注意，一定要记得在整数部分加上“0.”，因为十进制小数转换成二进制后仍是小数）。

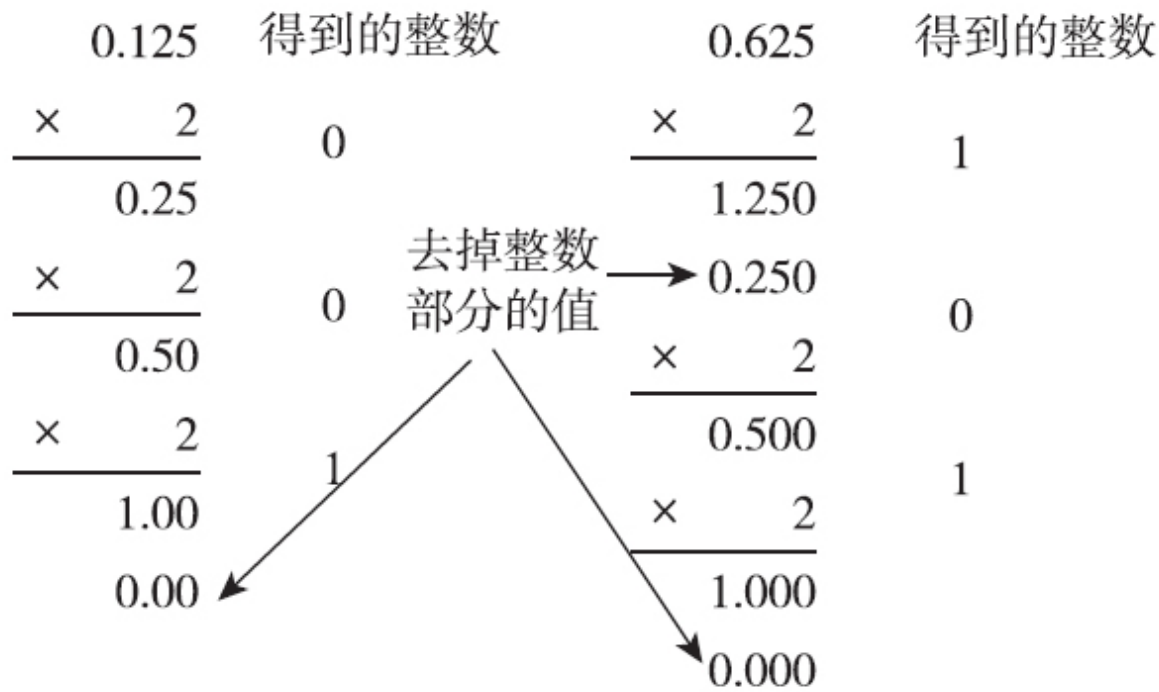


图 1-3 0.125和0.625十进制小数转换成二进制小数的步骤

注意 有些十进制小数乘以2后是个无穷循环数，永远不会有完整的整数，此时就要看所需的精度如何了，按所需位数精度取值即可。如0.825就是这样一个数，如果仅要求是小数点后3位，则相应的二进制

数为 $(0.110)_2$ ，如果要求为4位，则对应的二进制值为 $(0.1101)_2$ 。
具体如图1-4所示。

0.825	得到的整数
$\begin{array}{r} \times \quad 2 \\ \hline 0.650 \end{array}$	1
$\begin{array}{r} \times \quad 2 \\ \hline 0.300 \end{array}$	1
$\begin{array}{r} \times \quad 2 \\ \hline 0.600 \end{array}$	0
$\begin{array}{r} \times \quad 2 \\ \hline 0.200 \end{array}$	1

图 1-4 不同精确度要求的取值示例

如果一个十进制同时有整数和小数部分，则要对整数和小数部分分别按以上介绍的对应方法进行二进制转换。

2.十进制转换成八进制

八进制数的基数为8，因此八进制数中的数码有0、1、2、3、4、5、6、7，共八个。十进制转换成八进制的方法与前面介绍的十进制数转换成二进制的方法类似，只不过这里的基数是8（而不再是2）。十

进制转换成八进制当然也分整数部分和小数部分两种不同的转换方法。

十进制整数转换为八进制整数采用“除8逆序取余”的方法，直到所得的商小于8，然后把余数（包括最后一步中得到的小于8的商数）按逆序排列即可；十进制小数转换为八进制小数是采用“乘8正序取整”法，直到所得到的积小数部分为0，或者在规定的精度范围内，然后把所得到的整数正序排列起来即可。

图1-5左、上图是分别将十进制整数65和2467按“除8逆序取余”的方法转换成八进制的步骤，得到的结果分别是 $(101)_8$ 和 $(4643)_8$ 。

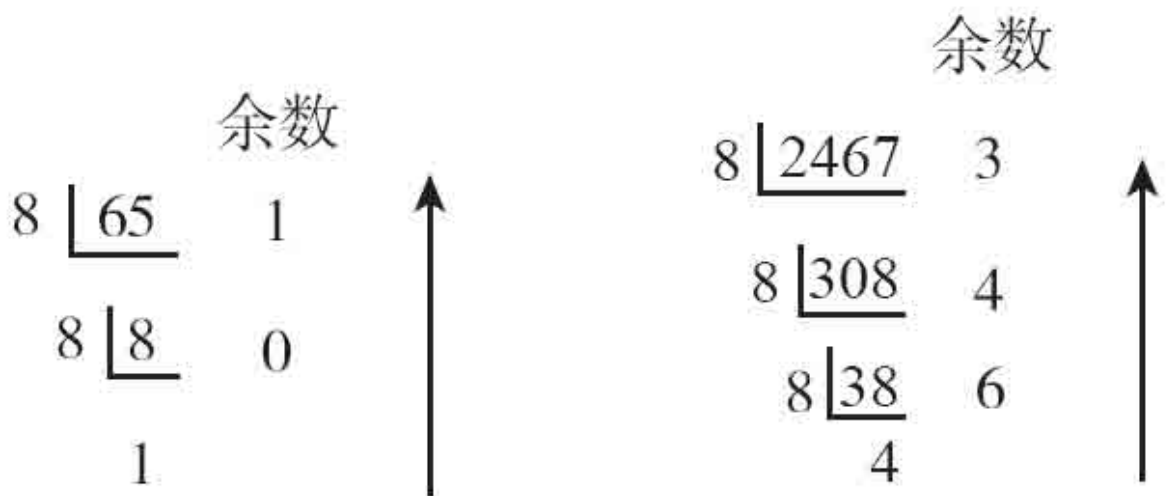


图 1-5 两个十进制整数转换成八进制整数的步骤

图1-6左、上图所示的是两个十进制小数0.125和0.8125通过“乘8正序取整”的方法转换成八进制的步骤，得到的结果分别是 $(0.1)_8$ 和

$(0.64)_8$ （是正序排列，一定要记得在整数部分加上“0.”）。

0.125	得到的整数	0.8125	得到的整数
$\begin{array}{r} \times \quad 8 \\ \hline 1.000 \\ 0.000 \end{array}$	1	$\begin{array}{r} \times \quad 8 \\ \hline 6.5000 \\ 0.5000 \end{array}$	6
	↓	$\begin{array}{r} \times \quad 8 \\ \hline 4.0000 \\ 0.0000 \end{array}$	4
			↓

图 1-6 两个十进制小数转换成八进制小数的步骤

3.十进制转换成十六进制

十六进制数的基数为16，十六进制数中的数码有0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F，共十六个。

十进制转换成十六进制与十进制转换成二进制类似，将十进制整数转换为十六进制的方法是采用“除16逆序取余”法，直到所得的商小于16，然后把余数（包括最后一步中得到的小于16的商数）按逆序排列即可；十进制小数转换为十六进制的方法是采用“乘16正序取整”法，直到所得到的积小数部分为0，或者在规定的精度范围内，然后把所得到的整数正序排列起来即可。

图1-7左、上图是分别将十进制整数45和3456按“除16逆序取余”的方法转换成十六进制的步骤，得到的结果分别是 $(2D)_{16}$ 和 $(D80)_{16}$ （注意，其中的13用十六进制的D表示了）。

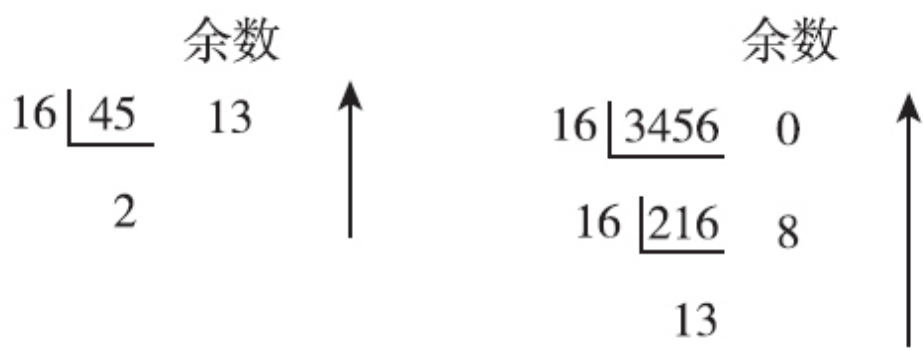


图 1-7 两个十进制整数转换成十六进制整数的步骤

图1-8左、上图是将十进制小数0.125和0.825通过“乘16正序取整”的方法转换成十六进制的步骤，得到的结果分别是 $(0.2)_{16}$ 和 $(0.D33)_{16}$ （精确到小数点后面三位）（注意：是正序排列，也一定要记得在整数部分加上“0.”，仍是小数）。

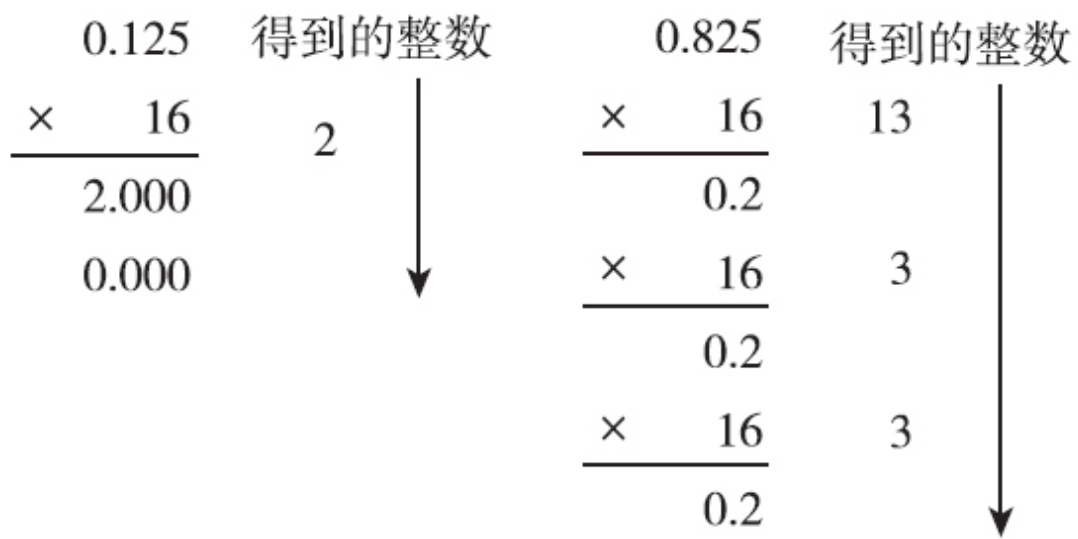


图 1-8 两个十进制小数转换成十六进制小数的步骤

4.同步练习

- 1) 把十进制数825、10815.6转换成二进制;
- 2) 把十进制数658、9240.65转换成八进制;
- 3) 把十进制数2508、5420.82转换成十六进制。

1.2.3 非十进制数之间的相互转换

从表1-1可以得出这样一个规律：1位八进制数对应3位二进制数，而1位十六进制数对应4位二进制数。因此，二进制数与八进制数之间、二进制数与十六进制数之间的相互转换十分容易。

1.相互转换方法

非十进制数间的具体转换方法如下：

1) 八进制数转换成二进制数的方法是：将每1位八进制数直接用相应的3位二进制来表示；二进制数转换成八进制数的方法是：以小数点为边界，整数部分向左，小数部分向右将每3位二进制分成一组，若不足3位则用0补足3位；然后将每一组二进制数直接用相应的1位八进制来表示。

例如要将 $(3456.2262)_8$ 转换为二进制数的方法是依次把八进制的每1位用3位二进制表示（如图1-9所示），最后的结果为 $(11100101110.010010101010)_2$ （整数部分最前面的0可以省略）。

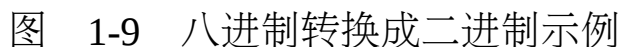


图 1-10 二进制转换成八进制示例

2) 十六进制数转换成二进制数的方法是：将每1位十六进制数直接用相应的4位二进制来表示。二进制数转换成十六进制数的方法是：以小数点为边界，整数部分向左，小数部分向右将每4位二进制数分成

一组，若不足4位则用0补足4位；然后将每一组二进制数直接用相应的1位十六进制表示。

例如将 $(4AF.51)_{16}$ 转换成二进制数的最后结果为 $(100\ 1010\ 1111.0101\ 0001)_2$ ；将二进制数 $(11\ 0110\ 1110.1010\ 1010\ 1000)_2$ （最后部分的三个0是划分位时补上去的，下同）转换为十六进制数的最后结果为 $(36E.AA8)_{16}$ 。

3) 八进制与十六进制的相互转换最好的方法就是先把其中一个转换成二进制，然后再把所得到的二进制转换成另一个进制的数。

如八进制数 $(6237.431)_8$ 转换成十六进制的步骤为：

步骤1：将 $(6237.431)_8$ 转换成二进制（每1位用3位二进制表示），得到的二进制结果是 $(110010011111.100011001)_2$ 。

步骤2：再将 $(1100\ 1001\ 1111.1000\ 1100\ 1000)_2$ 转换成十六进制（以小数点为边界，整数部分从右向左每4位分为一组，最后不足4位时加0补上，然后把小数部分从左向右将每4位二进制数分成一组，若不足4位则用0补足4位），得到最终的十六进制结果为 $(C9F.8C8)_{16}$ 。

将十六进制 $(3AB.11)_{16}$ 转换成八进制的步骤与上边的步骤类似，具体转换过程如下：

步骤1：先将十六进制数 $(3AB.11)_{16}$ 转换成二进制，得到 $(001110101011.00010001)_2$ 。

步骤2：再将 $(001\ 110\ 101\ 011.000\ 100\ 010)_2$ 转换成八进制以小数点为边界，整数部分从右向左每3位分为一组，最后不足3位时加0补上，然后把小数部分从左向右将每3位二进制数分成一组，若不足3位则用0补足3位，最后的结果为 $(1653.042)_8$ 。

以上二进制、八进制、十六进制的对应关系可参见表1-1，根据该表可直接进行转换。

2.同步练习

- 1) 把 $(1011011)_B$ 、 $(10110111.0010)_B$ 转换成八进制；
- 2) 把 $(1100111011)_B$ 、 $(11101001.101)_B$ 转换成十六进制；
- 3) 把 $(758)_O$ 、 $(8265.42)_O$ 转换成十六进制；
- 4) 把 $0xA58C$ 、 $0x8152.78$ 转换成八进制。

1.3 二进制数运算

二进制数在计算机中是应用最广的，因为它最简单，数码仅1和0两个，可以代表电平的高和低，或者电压的正和负，或者电路的开与关等。

对于二进制数，除了与十进制数一样可以进行四则算术运算外，还可以进行逻辑运算，因为它只有两个数码，可以代表两种截然相反的状态。本节将分别介绍二进制数的四则算术运算和逻辑运算。

1.3.1 二进制四则算术运算

二进制数的加、减、乘、除四则算术运算法则其实与十进制数的四则算术运算法则是一一对应的。理解了十进制数的四则算术运算法则，二进制数的四则算术运算就一点都不难了。下面分别予以介绍。

1.加、减法运算

首先要了解二进制的加、减法运算法则。

(1) 加法运算法则

$0+0=0$, $0+1=1$, $1+0=1$, $1+1=10$, 也就是当两个相加的二进制位仅一位为1时, 相加的结果为1; 如果两个二进制位全是0, 相加的结果仍为0; 而如果两个相加的二进制位均为1, 则结果为10 (相当于十进制中的2), 也就是“逢2进1”规则, 与十进制中的“逢10进1”的道理一样。

在进行二进制加减法运算时, 最关键的一点就是逢2进1, 进1当1, 而借1当2。大家联想一下我们经常使用的十进制数加法运算法则, 那就是逢10进1, 进1当1, 而借1当10, 这样一来我们就好理解了。二进制数的加法运算法则只是把原来十进制数加法运算法则中的10改成了2。

计算 $(10010)_2 + (11010)_2$ 的过程如图1-11所示 (注意两数要从最低位开始对齐)。

被加数	10010	}	这里被加数和加数一定要从 最低位开始一位位地对齐
加数	11010		
进位	1 1		
+			
	101100		

图 1-11 二进制加法运算示例

1) 首先是进行最低位相加，这里加数和被加数都为“0”，根据加法法则可以知道，相加后为“0”；

2) 再进行倒数第二位相加，这里加数和被加数都为“1”，根据加法法则可以知道，相加后为“ $(10)_2$ ”，此时把后面的“0”留下，而把第一位的“1”向高一位进“1”；

3) 再进行倒数第三位相加，这里加数和被加数都为“0”，根据加法法则可以知道，本来结果应为“0”，但倒数第二位已向这位进“1”了，此时就要同时把“被加数”、“加数”和“进位”这三个数加起来了，所以结果应为 $0+0+1=1$ ；

4) 再进行倒数第四位的相加，这里加数和被加数分别为“1”和“0”，倒数第三位也没有进位，根据加法法则可以知道，相加后为“1”；

5) 最后是最高位相加，这里加数和被加数都为“1”，根据加法法则可以知道，相加后为“ $(10)_2$ ”。同样需把第一位的“0”留下，并向高位进1，这样会产生新的最高位，值为“1”（如果超出了字长的限制，则新产生的最高位将溢出）。

这样 $(10010)_2 + (11010)_2$ 的最后运算结果为 101100 。

(2) 减法运算法则

1-1=0, 1-0=1, 0-0=0, 0-1=-1, 也就是当两个相加的二进制位中同为0或1时, 相减的结果为0; 如果被减数的二进制位为1, 而减数的二进制位为0, 则相减的结果仍为1; 而如果被减数的二进制位为0, 而减数的二进制位为1, 则需要向高位借1, 但此时是借1当2, 与十进制中的借1当10道理一样。

计算 $(111010)_2 - (101011)_2$ 的过程如图1-12所示 (注意两数要从最低位开始对齐)。

被减数	111010	}	这里被减数和减法一定要从最低位开始一位位地对齐
减数	101011		
借位	1111		
<div style="display: flex; align-items: center;"> <div style="width: 20px; border-bottom: 1px solid black; margin-right: 5px;"></div> <div></div> </div>			
	001111		

图 1-12 二进制减法运算示例

1) 首先是最低位相减, 这里被减数为“0”, 减数为“1”, 不能直接相减, 需要向高位 (此时为倒数第二位) 借“1”, 这样相当于得到了十进制中的“2”, 用2减1结果就得到1。

2) 再对倒数第二位相减, 此时本来被减数和减数均为“1”, 但是被减数的该位被上一步借走了1, 所以最后就变为“0” (1-1) 了。此时也不能直接与减数相减了, 又需要向高位 (此时为倒数第三位) 借1。

同样是借1当2，相当于该位总共为 $0+2=2$ 。这样倒数第二位相减后的结果为 $2-1=1$ 。

3) 用上一步同样的方法计算倒数第三位和倒数第四位的减法运算，结果都为1。

4) 再计算倒数第五位的减法运算，此时被减数原来为“1”，可是已被倒数第四位借走了1，所以成了“0”（ $1-1$ ），而此时减数也为“0”，可以直接相减，得到的结果为“0”。

5) 最后计算最高位的相减，被减数和减数均为“1”，可以直接相减，得到的结果为“0”。

这样一来， $(111010)_2 - (101011)_2$ 的结果是 $(001111)_2$ ，由于整数的前导0可以不写，所以最后结果就是 $(1111)_2$ 。

2.乘、除法运算

同样，首先要了解二进制的乘、除法运算法则。

(1) 乘法运算法则

$0 \times 0 = 0$ ， $0 \times 1 = 0$ ， $1 \times 0 = 0$ ， $1 \times 1 = 1$ ，也就只有当两个相乘的二进制位都为1，相乘的结果才为1；两个相乘的二进制位中只要有一位为0（也包括是两位同时为0），则相乘的结果都为0。也可以这么理解：1与任

何数相乘的结果就是对应的被乘数；而0与任何数相乘的结果都为0。这与十进制中的乘法运算法则也是一样的。要注意的是，在乘法运算中，乘数的每一位要与被乘数的每一位分别相乘，而不仅是对应位相乘，而且乘数的每一位与被乘数的每一位相乘的结果的最低位要与对应的乘数位对齐。当然这与十进制的乘法运算法则也是一样的，很好理解。

计算 $(1010)_2 \times (101)_2$ 的过程如图1-13所示（注意两数要从最低位开始对齐）。

被乘数	1010	
乘数	101	
×		
<hr/>		
	1010	} 这里每行的最低位一定要和对应的乘数位对齐
	0000	
+	1010	
<hr/>		
积	110010	

图 1-13 二进制乘法运算示例

1) 首先是乘数的最低位与被乘数的所有位相乘，因为乘数的最低位为“1”，根据乘法法则可以得出，结果其实就被乘数本身，直接复制下来即可。此处结果就为1010。

2) 接着进行的是乘数的倒数第二位与被乘数的所有位相乘, 因为乘数的这一位为“0”, 根据乘法运算法则可以得出, 结果均为“0”。此处结果就为0000。

3) 然后是乘数的最高位与被乘数的所有位相乘, 此时乘数为“1”, 结果就是被乘数本身。此处结果就为1010。

4) 最后再按照前面介绍的二进制加法原则对以上三步所得的结果按位相加 (但这里的位对齐方式与单纯的二进制数相加不一样, 最低位一定要与对应乘数位对齐。这也与十进制的乘法运算方法一样), 结果得到 $(110010)_2$ 。

(2) 除法运算法则

当被除数大于除数时, 商是“1”; 当被除数小于除数时, 不够除, 商只能是“0”, 这与十进制的除法也类似。二进制只有两个数 (0, 1), 因此它的商也只能是1或0。

计算 $(11001)_2 \div (101)_2$ 的过程如图1-14所示。

$$\begin{array}{r}
 \text{商 } 101 \\
 \text{除数 } 101 \overline{) 11001} \quad \text{被除数} \\
 \underline{- 101} \\
 101 \\
 \underline{- 101} \\
 000
 \end{array}$$

图 1-14 二进制除法运算示例

1) 因为除数为“101”，有3位，所以在被除数中也至少要取3位（从最高位开始取）。被除数的高3位为“110”，恰好比除数“101”大，可以直接相除，但商只能是1（因为二进制的最大数元就是1），然后把被除数减法商“1”与除数相乘后的结果，得到的值为“1”。

2) 再从被除数中取下一位“0”下来，与上一步的差“1”值组成新的被除数，为“10”，显然它比除数“101”小，不够除。于是在商的对应位置上输入“0”。

3) 继续从被除数中取下一位“1”下来，与上一步的余数“10”值组成新的被除数，为“101”，此数正好与除数“101”相等，所以此时的商取“1”，正好除尽。

这样一来 $(11001)_2 \div (101)_2$ 所得的商就是 $(101)_2$ 了。

3.同步练习

- 1) 求 $(011101)_B + (10010)_B$ 、 $(100111)_B + (110110)_B$ 的值;
- 2) 求 $(1110101)_B - (110010)_B$ 、 $(1101011)_B - (10001)_B$ 的值;
- 3) 求 $(1110)_B \times (1001)_B$ 、 $(1100)_B \times (10111)_B$ 的值;
- 4) 求 $(110010)_B \div (1010)_B$ 、 $(100110101011)_B \div (1011)_B$ 的值。

1.3.2 二进制逻辑运算

逻辑运算是指对因果关系进行分析的一种运算，这也是在计算机中经常采用的一种二进制运算。逻辑运算的结果并不表示数值大小，而是表示一种逻辑概念，若成立则为“真”，或用“1”表示；若不成立，则为“假”，或用“0”表示。二进制的逻辑运算主要有“与”、“或”、“非”和“异或”四种。

1.“与”运算 (AND)

“与”运算又称逻辑乘，用符号“.”或“ \wedge ”来表示。运算法则如下：

$$0 \wedge 0 = 0 \quad 0 \wedge 1 = 0 \quad 1 \wedge 0 = 0 \quad 1 \wedge 1 = 1$$

归纳起来也就是在“与”运算中，只要两个参加“与”运算的数的对应位有一个为0，运算结果就为0；仅当两数的对应位均为1时结果才为1，很容易判断。这与前面介绍的二进制乘法运算是一样的。图1-15所示是两个“与”的逻辑运算示例。图1-15a所示的是两个位数不一样的二进制数进行“与”运算，这时要求两个数从最低位开始对齐，在位数少的二进制的最高位前面加上“0”补齐，使得它与位数多的二进制数有一样的位数。

<p>补的“0” 10011</p> <p> \swarrow</p> <p> 01001</p> <p>\wedge</p> <hr style="width: 100px; margin: 0 auto;"/> <p> 00001</p> <p style="text-align: center;">a)</p>	<p>11110110</p> <p>\wedge 11011101</p> <hr style="width: 100px; margin: 0 auto;"/> <p>11010100</p> <p style="text-align: center;">b)</p>
---	--

图 1-15 两个“与”逻辑运算示例

2.“或”运算 (OR)

“或”运算又称逻辑加，用符号“+”或“ \vee ”表示。运算法则如下：

$$0\vee0=0 \quad 0\vee1=1 \quad 1\vee0=1 \quad 1\vee1=1$$

也就是说，在“或”运算中，只要两个参加“或”运算数的对应位中有一个为1，运算结果就为1，仅当两数的对应位均为0时结果才为0，也很容易判断。如图1-16所示是两个“或”的逻辑运算示例。同样，进行“或”运算时要求两数从最低位开始对齐，位数少的数在最高位前面加“0”补齐，最终使两个二进制数的位数相同。

<p>补的“0” 1001110</p> <p> \swarrow</p> <p> 0110110</p> <p>\vee</p> <hr style="width: 100px; margin: 0 auto;"/> <p> 1111110</p> <p style="text-align: center;">a)</p>	<p>100101010</p> <p>\vee 100111100</p> <hr style="width: 100px; margin: 0 auto;"/> <p>100111110</p> <p style="text-align: center;">b)</p>
---	---

图 1-16 两个“或”逻辑运算示例

3.“非”运算 (NOT)

“非”运算就是逐位求反的运算，其运算法则为：“0”的反值为“1”，“1”的反值为“0”，也就是“0”与“1”互为反。注意：“非运算”只是针对一个二进制数进行的，这与前面的“与”和“或”运算不一样。如“101110101”进行“非”运算后就得到“010001010”（可简写为“10001010”）。

4.“异或”运算 (XOR)

“异或”运算用符号“ \oplus ”来表示。其运算法则如下：

$$0 \oplus 0 = 0 \quad 0 \oplus 1 = 1 \quad 1 \oplus 0 = 1 \quad 1 \oplus 1 = 0$$

也就是说，当两个参加“异或”运算的二进制数对应位相同时运算结果为0，不同时运算结果为1。图1-17所示是“异或”逻辑运算示例。同样，进行“异或”运算时要求两数从最低位开始对齐，位数少的数在最高位前面加“0”补齐，最终使两个二进制数的位数相同。

<div style="display: flex; align-items: center;"> <div style="margin-right: 10px;">补的“0” ⊕</div> <div style="text-align: right;"> $\begin{array}{r} 1001110 \\ 0110110 \\ \hline 1111000 \end{array}$ </div> </div> <p style="text-align: center;">a)</p>	<div style="text-align: right;"> $\begin{array}{r} 100101010 \\ 100111100 \\ \hline 000010110 \end{array}$ </div> <p style="text-align: center;">b)</p>
--	--

图 1-17 两个“异或”逻辑运算示例

5.同步练习

- 1) 求 $(11001)B \wedge (1011)B$ 、 $(10011)B \wedge (10101)B$ 的结果;
- 2) 求 $(11001)B \vee (1011)B$ 、 $(10011)B \vee (10101)B$ 的结果;
- 3) 求 $(1100110)B$ 、 $(1000111)B$ 非的结果;
- 4) 求 $(1100110)B \oplus (1011)B$ 、 $(100111)B \oplus (10101)B$ 的结果。

1.4 二进制数的表示形式

结束了以上各种进制数之间的转换，我们来谈谈另一个话题：原码、反码、补码和移码。其实它们都是计算机中二进制数（称之为“机器数”）的不同表示形式，之所以会有这么多码，最根本的原因就是计算机只有加法运算器，减法运算需要转换成负数的加法。这里又出现一个问题，那就是计算机如何来识别负数？因为它不能像人的大脑一样可以识别“+”、“-”之类的符号，故必须用一个专门的位来表示数的符号，这就是计算机中的“有符号数”的由来。

我们已经知道，计算机中所有数据最终都是使用二进制数表示的。我们也已经学会如何将一个十进制、八进制和十六进制数转换为二进制数，但我们仍然不知道如何将一个负数用二进制形式来表示，因为我们前面学习的都是针对正数的，无论是整数，还是小数。其实，在计算机中负数是以其正数的补码形式表示的，这就是本节所要介绍的主题。

1.4.1 二进制数的真值和字长

前面说了，负数是通过其正数的补码形式来表示的，那什么是补码呢？这还得从原码、反码说起。在计算机二进制数中其实也包括符

号位，并不是所有位都代表数据本身，就像我们常用的十进制数中有正、有负一样。但是二进制中没有“-”这个符号，那么计算机中的机器数的符号是怎样规定的呢？这得从机器数的两个基本概念（真值和字长）说起。

（1）真值

计算机中的二进制机器数分为“有符号数”和“无符号数”两种。“无符号数”就是二进制数的每一位都代表对应位的数值；而在“有符号数”中规定最高位用来表示数据符号，其中1代表负，0代表正，这样一来机器数本身就不等于真正的数值了。例如有符号数10000101，其最高位1代表负，所以余下的“0000101”才是数值本身，所以其真正数值是-5，而如果是无符号数，则10000101所代表的是133。为区别起见，把带符号位的机器数所对应的真正数值称为机器数的“真值”。例：
00100001的真值=0 0100001=+33（正号可以不写，可以直接写成33），10100011的真值=1 0100011=-35。

（2）字长

在机器数中还有一个概念需要首先弄明白，那就是“字长”。“字长”是指计算机一次可处理的二进制数的码位长度，是计算机进行数据存储和数据处理的运算单位。如我们通常所指的32位处理器，就是指

该处理器的字长为32位，也就是一次能处理32位二进制数。通常称16位是一个字，32位是一个双字，64位是两个双字。

数值的转换结果是与字长有关的。如果计算机字长为8位，十进制中的数+5转换成二进制就是00000101，-5转换成二进制就是10000101；但如果字长是16位，+5转换的结果就是00000000 00000101，而-5转换成二进制就是10000000 00000101了。也就是对应的机器数要转换为字长所代表的位数。

字长越长代表计算机的处理能力越强，可以处理的数越大。如现在字长普遍是64位的了，这样一来计算机可以处理的二进制数码位长度最大为64位，去掉符号位，则表示计算机可以处理的最大二进制数为2的63次方，最小二进制数就是-2的63次方。而在8位字长中，因为最高位要用于符号位，所以实际可以处理的数值大小范围为-127~-0~0~127（即 $2^7 - 1$ ），共256个数了；但如果字长是16位，可以处理的数值大小就可以是-32767~-0~0~32767($2^{15} - 1$)。

注意 以上的“-0”与“0”的机器数是不一样的，在8位字长中，-0为1 0000000，而+0为0 0000000；在16位字长中，-0为1 0000000 00000000，而+0为0 0000000 00000000。所以在二进制的机器数中，0也有两个（-0和0），且表示形式并不一样。

1.4.2 二进制数的四种表示形式

计算机中的二进制数有四种主要表示形式，那就是原码、反码、补码和移码，其中最重要的是前三种。反码和补码是为了在二进制中表示负数而产生的。本节具体介绍这四种表示形式。要注意的是，采用哪种表示形式，运算的结果也是对应的表示形式。

1.原码

对于人脑来说，我们都知道，+表示正数，-表示负数，然而在计算机二进制中也引入这两个符号肯定是不行的，因为在计算机中只有0和1这两个字，根本不认识“+”和“-”这两个符号。计算机中的任何行为都依赖于它的物理结构。计算机是没有思维的，所以得让计算机在0和1之间识别出对应数的正与负。

最开始的时候，人们约定在一个二进制数前用第一位（最高位）来表示符号，即1表示负，0表示正，这就是最初“原码”的概念。“原码”就是“原始码位”，或者“原始编码”的意思，就是对应二进制数本身所代表的形式。比如，+3（以8位字长为例，下同），符号位为0，3转化为二进制就是11，那么+3的原码就是00000011（最高位为符号位，正数的符号位为“0”，其余数值位不足部分补0）。同理，-3的符号位为1，3转化为二进制就是11，最终-3的原码就是10000011（不足8

位时在前面用0补足)。在日常的书写中，原码的表示形式是用方括号下面加上一个“原”字下标来区别的，如 $[+3]_{\text{原}} = 00000011$ ， $[-3]_{\text{原}} = 10000011$ 。

再来计算+127和-127各自的原码。+127中符号位为“0”，127的二进制为“1111111”，这样 $[+127]_{\text{原}} = 01111111$ ；而-127中的符号位也为“1”，所以最终 $[-127]_{\text{原}} = 11111111$ 。

最应该注意的是，在原码表示形式中，0有“+0”和“-0”之分。对应的原码分别是0 0000000和1 0000000。

2.补码

原码的设计很不错，至少可以成功地区分出二进制数的正与负了，但是这种方法仍有一些局限性，那就是原码在加、减法运算中不方便，符号位需要单独处理、单独判断。同为正数的加、减是没什么问题的，可是异号相加、减时就存在问题了。如1-1，如果用原码计算的话结果为-2，显然不正确。

$$(0\ 0000001)_{\text{原}} + (1\ 0000001)_{\text{原}} = (1\ 0000010)_{\text{原}} = -2$$

另外，在原码中0有+0和-0两种表示形式，这就存在二义性了，在计算机中是绝对不能容忍的。于是后来就想到一种能解决原码中存在的这些问题的另一种表示形式——补码。

补码的编码规则如下：正数的补码和原码相同；负数的补码是通过先把除符号位外其他各位取反，再在末位（最低位）加1得到的。这样，我们只要让减数通过一个求反电路，再通过一个+1电路，然后再通过加法器就可以实现减法运算了。

经验之谈 如果要把一个补码转换成原码该如何操作呢？很简单，只需要把原码转换成补码的过程倒过来操作就行了。因为正数的补码与原码一样，所以正数补码的原码就是其补码，而负数补码的原码是先最后在最后一位加1，然后对其除符号位外的其他各位全部取反得到。只有相同码制的数才能进行操作，结果就是对应的码制，也就是原码数与码数的运算，结果也为原码，反码数与反码数的运算结果也为反码，补码数与补码数的运算结果也为补码。如果结果是负数，要判断结果是否正确，需要再将其对应的码制转换为原码。

补码首先继承了原码的特点（可以表示正与负），而且它包括了两个优点：

□可以把符号位一起运算；

□0只有一种表示形式，没有二义性。

下面同样以上面的1-1为例，如果采用补码形式，则算式如下：

$$(0\ 0000001)_{\text{补}} + (1\ 1111111)_{\text{补}} = 0\ 0000000 = 0$$

结果完全正确。其实这里涉及一个计算机运算中“模”的概念。

我们把一个计量单位称为模或模数。例如，时钟是以十二进制进行计数循环的，即以12为模。在时钟上，时针加上（正拨）12的整数倍或减去（反拨）12的整数倍，时针的位置不变。例如，14点钟在舍去模12后，成为（下午）2点钟（ $14=14-12=2$ ）；从0点出发逆时针拨10格即减去10小时，也可看成从0点出发顺时针拨2格（加上2小时），即2点（ $0-10=-10=-10+12=2$ ）。因此，在模12的前提下，-10可映射为+2。

由此可见，对于一个模数为12的循环系统来说，加2和减10的效果是一样的。所以，在以12为模的系统中，凡是减10的运算都可以用加2来代替，这样就把减法问题转化成加法问题了（注：计算机的硬件结构中只有加法器，所以大部分的运算都必须最终转换为加法）。10和2对模12而言互为补数。

同理，计算机的运算部件与寄存器都有一定字长的限制，因此它的运算也是一种模运算。如果字长为8，则当计数器计满8位也就是256个数后会产生溢出，又从头开始计数。产生溢出时的那个量就是计数器的模。显然，8位二进制数的模数为2的8次方，即256。在计算中，两个互补的数称为“补码”，这就是“补码”表示形式诞生的由来。

十进制数 $2-3=(0\ 0000010)_{\text{补}}+(1\ 1111101)_{\text{补}}=(11111111)_{\text{补}}=-1$ ，也是正确的。十进制数 $123-121=(0\ 1111011)_{\text{补}}+(1\ 0000111)_{\text{补}}=(0\ 0000010)_{\text{补}}=2$ ，也是正确的。

另外，在补码表示形式中，0仅有一种表示形式，因为无论是“+0”，还是“-0”的补码均为0 0000000。

3.反码

通过以上介绍，我们可以知道，补码是我们的最佳选择，因为它可以全面解决异号二进制数之间的加减运算问题。那为什么还有“反码”这种表示形式呢？其实“反码”是“原码”向“补码”表示形式转变过程中的一个过渡形式，最终证明它是失败的。之所以当初会想到“反码”，是因为它太容易从电路上来实现了（仅需要取反就行了）。但“反码”专门是针对负数进行的（正数的反码与原码一样），就是对二进制负数按位（除符号位外）取反，原来为1就变为0，原来为0就变为1。而且采用反码形式对于一些异号二进制运算还是正确的，如上面所说的“2-3”，采用反码运算的格式如下：

$$(0\ 0000010)_{\text{反}}+(1\ 1111100)_{\text{反}}=(1\ 1111110)_{\text{反}}=-1$$

但是在1-1的反码运算中，结果就不正确了：

$$(0\ 0000001)_{\text{反}}+(1\ 1111110)_{\text{反}}=(1\ 1111111)_{\text{反}}=-0$$

本来应该是+0的，结果却成了-0。

再如十进制数“123-121”，用反码加法运算就得到：

$$(0\ 1111011)_{\text{反}} + (1\ 0000110)_{\text{反}} = (0\ 0000001)_{\text{反}} = 1$$

显然也不正确。

注意 反码是相互的，如10011001的反码为11100110（符号位是不变的），相反11100110的反码也是10011001。另外，与原码一样，在反码表示形式中0也有“+0”和“-0”之分，对应的反码分别为0 0000000和1 1111111。

综上所述可以得出：正数的原码、反码和补码都是一样的，而负数的这三种表示形式就不一样了，负数的反码是对原码中除符号位外的其他各位取反，而负数的补码是再对其反码加1，也就是先对其原码中除符号位外的其他各位取反，然后再在最低位加1。

4.移码

移码是一种比较特殊的二进制数表示形式。它的编码规则如下：

□正数的符号位为1，负数的符号位为0；

□真值部分与补码一样。

从以上两个编码规则中可以看出，要求一个二进制的移码，只需先求出它的补码，然后再把符号位取反（因为在补码中规定符号位1表示负数，0为正数，正好相反）就行了，所以移码又称为符号位取反的补码。如：5的二进制移码（假设字长为8）为10000101，-5的二进制移码为01111011（它是对-5的补码符号取反的结果）。

其实还有一种更简单的计算移码的方法，那就是 $[X]_{\text{移}} = 2^n + X$ （ n 为二进制数的位数，不包括符号位）的二进制值+ X 。如 $[-11010]_{\text{移}} = [2^5]_{\text{B}} + (-11010) = 1000000 - 11010 = 000110$ 。现在我们再来利用上面所介绍的补码符号位取反的方法来重新计算一下，验证以上结果是否正确。

$[-11010]_{\text{B}}$ 的补码为100110，符号位取反后得到000110，最后证明两种算法结果一样。

1.4.3 补码的加减法运算

上节介绍了原码、反码、补码和移码的转换方法，同时我们知道，在计算机机器数中实际上全是采用补码方式进行运算的，特别是减法运算。因为原码和反码的减法运算有时结果是不正确的。所以本节仅介绍补码的加、减法运算方法。

1.补码的转换

在正式介绍补码的加减法运算前，还是先回顾一下补码的运算方法，这是补码运算的基础。通过前面的学习，我们知道，机器数的补码可由原码和反码得到。如果机器数是正数，则该机器数的补码与原码一样；如果机器数是负数，则该机器数的补码是对它的反码在末位加1而得到的。

例如，当 $X=+0.1011$ 时，根据以上规则可得到 $[X]_{\text{补}}=0.1011$ （因为正数的补码与原码、补码一样）。而当 $X=-0.1011$ 时，则 $[X]_{\text{补}}=1.0101$ （负数的补码是符号位不变，真值是在它的反码基础上最后位加1得到的）。这里的“1011”的反码（也就是按位取反）为“0100”，再在末位加“1”后即得到了“0101”，最后再加上符号位“1”，所以最后的值为1.0101（符号位是最高位）。

又例如， $X=+1010$ 时，则 $[X]_{\text{补}}=01010$ （正数的补码与原码、反码一样，注意在最高位要体现数的正、负符号）。而当 $X=-1010$ 时，则 $[X]_{\text{补}}=10110$ 。运算方法一样：先计算“1010”的反码，为“0101”，然后再在末位加“1”，得到“0110”，最后再加上符号位“1”，即得到“10110”。

经验之谈 整数“0”的补码只有一种表示形式，即00...0，因为“-0”的补码最终结果也是“0”，如字长为8时，-0的原码就为10000000，求其补码时，首先是在原码中除符号位外的其他位取反（得到1 1111111），然后在最后位加1，得到10 0000000，注意这里一共有9位了。因为字长为8位，所以9位的1会溢出，这样最后得到的补码值同样为“00000000”，也就是0。如果采用16位字长，“-0”的补码同样为“00000000 00000000”，因为这将产生值为“1”的第17位，而这个第17位在16位字长中同样是溢出的。

表1-2是8位二进制的原码、反码、补码对照表。在进行表示形式转换时如果记得这个数，有时速率会快许多。

表 1-2 8 位二进制的原码、反码、补码对照表

二进制数码	对应的无符号数	对应的有符号数的原码	对应的有符号数的反码	对应的有符号数的补码
00000000	0	+0	+0	+0
00000001	1	+1	+1	+1
00000010	2	+2	+2	+2
⋮	⋮	⋮	⋮	⋮
01111111	126	+126	+126	+126
01111111	127	+127	+127	+127
10000000	128	-0	-127	-128
10000001	129	-1	-126	-127
10000010	130	-2	-125	-126
⋮	⋮	⋮	⋮	⋮
11111110	254	-126	-1	-2
11111111	255	-127	-0	-1

2.补码的加法运算

补码的加法运算法则如下：

$$[X+Y]_{\text{补}} = [X]_{\text{补}} + [Y]_{\text{补}}$$

该式表明，两个有符号数相加的补码可以通过先分别对两个数求补码，然后相加得到。在采用补码形式表示时，进行加法运算时可以把符号位和数值位一起进行运算（若符号位有进位，则溢出不管），结果为两数之和的补码形式。

如要求两个十进制数：35+18的补码（假设字长为8）。根据上面的补码加法运算法则可以得知，只需分别求35和18这两个数的补码，然后相加即可。又因这两个数都是正数，所以它们的补码与原码一样。这样一来，这道题实际上也就是求35和18这两个十进制数的原码

和。35的原码为0 0100011（注意：最高位为符号位），18的补码为0 0010010。所以35+18的补码就等于(0 0100011)B+(0 0010010)B=(00110101)B，如图1-18a所示。如果转换成十进制就等于53，结果正确。如果相加后有超过字长的位溢出，则直接丢弃。

同理，如果要求两个十进制数：35+(-18)和的补码也是直接求35和-18的补码和。35的补码与其原码一样，前面已计算出，为0 0100011；而后面那个“-18”因为是负数，所以不能直接从它的原码得到补码。需要先求-18的原码1 0010010，然后对其除符号位外的其他各位取反，得到其反码（为1 1101101），最后再在其末位（最低位）加1，最终得到其补码为1 1101110。

这样一来，“35-18”的补码就是(00100011)B+(11101110)B，结果为00010001，如图1-18b所示。这里要注意，两个补码相加后产生了第9位（为1）的溢出，直接丢弃，所以结果就是(00010001)B。如果转换成十进制的话就等于17，结果正确。

$ \begin{array}{r} 0\ 0100011 \\ +\ 0\ 0010010 \\ \hline 0\ 0110101 \end{array} $	$ \begin{array}{r} 0\ 0100011 \\ +\ 1\ 1101110 \\ \hline 1\ 0\ 0010001 \end{array} $
a)	<div style="display: flex; align-items: center; justify-content: center;"> <div style="text-align: right; padding-right: 10px;"> 新产生的最 高位，溢出 </div> <div style="font-size: 2em;">↗</div> </div> b)

图 1-18 两个补码加法运算示例

3.补码的减法运算

补码的减法运算法则如下：

$$[X-Y]_{\text{补}} = [X]_{\text{补}} + [-Y]_{\text{补}}$$

该公式表明，求两个机器数的差值（如 $[X-Y]_{\text{补}}$ ）的补码，可以通过求被减数的补码（如 $[X]_{\text{补}}$ ）与减数的负值的补码（ $[-Y]_{\text{补}}$ ）的和得到。

$[-Y]_{\text{补}}$ 是对减数进行求负操作，求负的规则是全部位（含符号位）取反后再加1（实际上也是分别对符号位和真值位进行求反，因为正数与负数的符号也正好相反）。例如：已知 $[15]_{\text{补}} = 00001111$ ，则 $[-15]_{\text{补}} = 11110000 + 1 = 11110001$ 。

现在假设 $X=+35$ ， $Y=+18$ ，要求 $[X-Y]_{\text{补}}$ （字长为8）。

先根据正数的补码与原码一样的规则，求得 $[X]_{\text{补}} = 00100011$ ， $[Y]_{\text{补}} = 00010010$ ；再根据以上介绍的补码求负操作规则，即可得到 $[-Y]_{\text{补}} = 11101110$ ；最后用 $[X]_{\text{补}} + [-Y]_{\text{补}}$ 公式即可得到最终的 $[X-Y]_{\text{补}} = 00010001$ ，如图1-19a所示。转换成十进制，也可得到结果17，正确，且与上面使用加法法则运算的结果一样。注意，这里相加的结果也产生了溢出的第9位（1），直接丢弃。

$ \begin{array}{r} 0\ 0100011 \\ +\ 1\ 1101110 \\ \hline 1\ 0\ 0010001 \end{array} $	$ \begin{array}{r} 1\ 1011101 \\ +\ 1\ 1101110 \\ \hline 11\ 1001011 \end{array} $
新产生的最 高位，溢出 a)	新产生的最 高位，溢出 b)

图 1-19 两个补码减法运算示例

注意 原码、反码和补码运算的结果也是对应的表示形式，因为正数的原码、反码和补码都一样，所以当反码、补码的运算结果为正数时，反码和补码的结果也就是对应的原码。但是如果结果是负数，则反码、补码的结果不等于原码，必须经过相应的操作才能转换为原码。如负数的反码要转换为原码必须对反码除符号位外的其他位全部取反；而负数的补码要转换为原码必须先在该补码的最后一位减1，然后对除符号位外的其他各位取反才能得到。

如有两个十进制数分别是 $X=-35$ ， $Y=-18$ ，现要求 $[X+Y]_{\text{补}}$ 的值，很显然这个结果是一个负数。根据前面介绍的加法法则得知，可先求得 $[X]_{\text{补}}$ 和 $[Y]_{\text{补}}$ 的值，然后再相加，即 $[X]_{\text{补}}+[Y]_{\text{补}}$ 。根据本章前面介绍的知识，我们很快可以算出 $[-35]_{\text{补}}$ 为1 1011101， $[-18]_{\text{补}}=11101110$ 。最后 $[-35]_{\text{补}}+[-18]_{\text{补}}$ 的运算过程如图1-19b所示，结果为11001011。注意这是一个有符号位数，所以结果为-75。这样看起来结果是不正确的，因为-35-18应该等于-53。这时就要特别注意了，这个-75是一个补码形

式，要看最终的结果，还得把它转换成原码。按照上面介绍的方法可得 $[-75]_{\text{补}}$ 的原码为10110101（注意它也是一个有符号位，最高的1代表为负数）= $[-53]_{\text{原}}$ 。这样结果就正确了。

4.同步练习

- 1) 求 $[85+24]_{\text{补}}$ 、 $[152+35]_{\text{补}}$ 的值；
- 2) 求 $[185-56]_{\text{补}}$ 、 $[52-135]_{\text{补}}$ 的值。

第2章 计算机网络概述

本章作为本书的开篇（除了作为选学内容的第1章），先从宏观角度对计算机网络进行了概括性的介绍，以便使大家对计算机网络有一个基本的了解。本章内容比较丰富，这对于以前对计算机网络方面的知识了解较少的朋友来说非常重要。相信你学完本章之后，再与朋友谈论有关计算机网络方面的话题时也可以对上几句了，再也不会像以前一样被“冷落”，让朋友觉得你完全是一个局外人了。

本章主要介绍计算机网络的一些基础知识，包括计算机网络的发展历史、基本组成、作用、分类、拓扑结构等。本章的重点是理解各种拓扑结构的主要特性和优缺点。当然，本章只是概述，详细介绍将在后续章节进行。

2.1 计算机网络概述

学习计算机网络，应从了解计算机网络定义、计算机网络发展历史、计算机网络的基本组成和主要应用开始。这些都是最基本、最基础的计算机网络的知识，也是我们平时与朋友经常聊到的内容。

2.1.1 计算机网络的定义

“计算机网络定义”就是“什么是计算机网络”。其实这个问题并不是很重要，而且也没有一个非常精确的定义。一开始只要我们知道眼前所见到的那些由许多计算机设备通过电缆连接在一起所组成的系统就是计算机网络就行了。

计算机网络从诞生起，发展至今也仅有50多年的历史了（真正的计算机网络是从1969年美国国防部高级研究计划署（ARPA）建立ARPANet开始的，这部分内容将在本章后面介绍）。在这短短的50来年的时间里，各方面的计算机网络技术的发展和普及真可谓“日新月异”，真的很难想象最早的几兆连接速率在这么短时间里已发展到了几十吉的连接速率，增加了几千倍。随着计算机网络技术和应用的不断发展，计算机网络的内涵也在不断发生变化，所以关于“计算机网络”，至今仍没有一个严格意义上的权威定义。

目前通常认为“计算机网络”是指将不同地理位置，具有独立功能的多台计算机及网络设备通过通信线路（包括传输介质和网络设备）连接起来，在网络操作系统、网络管理软件及网络通信协议的共同管理和协调下实现资源共享和信息传递的计算机系统。如果你还不知道计算机网络为何物的话，回到家里，或者到网吧，或者去你所在的公司去亲眼看一下吧。你所看到的那一台台看似独立、位于不同位置的PC（个人计算机），通过一些电缆和一些盒子状的设备（如交换机、路由器）连接起来的就是一个计算机网络。

简单地讲，计算机网络就是许多独立工作的计算机系统通过通信线路（包括连接电缆和网络设备）相互连接构成的计算机系统集合，或者计算机系统团体。而在这个计算机系统集合中，可以实现各计算机间的资源共享、相互访问，可以进行各种需要的计算机网络应用。其中的计算机可以是微机、小型机、中型机、大型机或巨型机等，网络设备包括网桥、网关、交换机、AP、路由器、防火墙等。但仅有这些硬件是不可能组成计算机网络的，还必须有相应的软件系统支持，这方面内容将在本章后面介绍计算机网络组成时再详述。这里所说的“资源共享”包括：硬件资源共享、软件资源共享、数据资源共享这三个方面。硬件资源共享的最典型例子就是我们最常用的打印机共享（还记得和其他电脑使用一台打印机来打印文件的情形吗？如图2-1所示）、磁盘共享（如网吧中的无盘网络）。

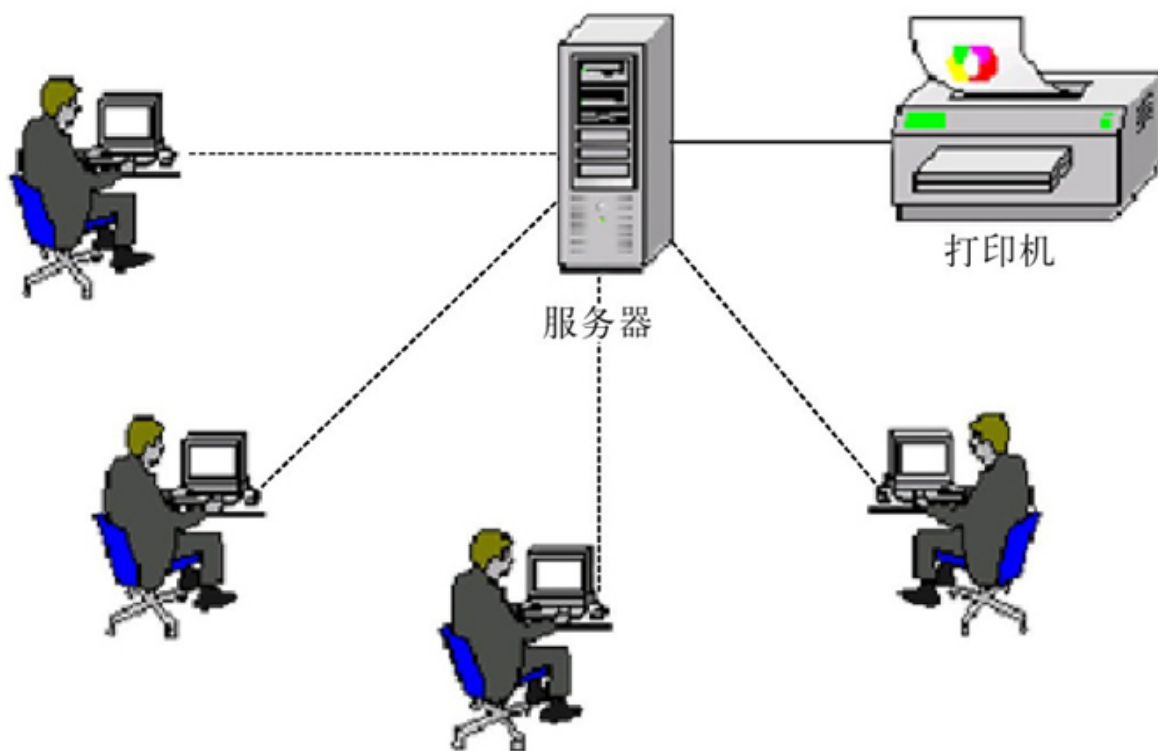


图 2-1 硬件资源共享示例

数据资源共享的典型例子就是数据库资源共享，各网络用户可以集中调用一台数据库服务器中的相关数据信息，如图2-2所示。各种应用服务器也是数据资源共享的实例，如通过FoxMail、OutLook等客户端邮件软件收取邮件，你和你的好友天天玩的网络游戏，或者你和你的家人天天在家里不同电脑中看的网上的同一部电影等。至于软件资源共享的例子也是非常多的，比如在企业内部网络中我们会在服务器上为所有员工提供一些常用工具软件共享，让用户自己选择安装。如果推到互联网上，则更直观了，我们从网上下载的各种软件都是软件资源共享的实例。

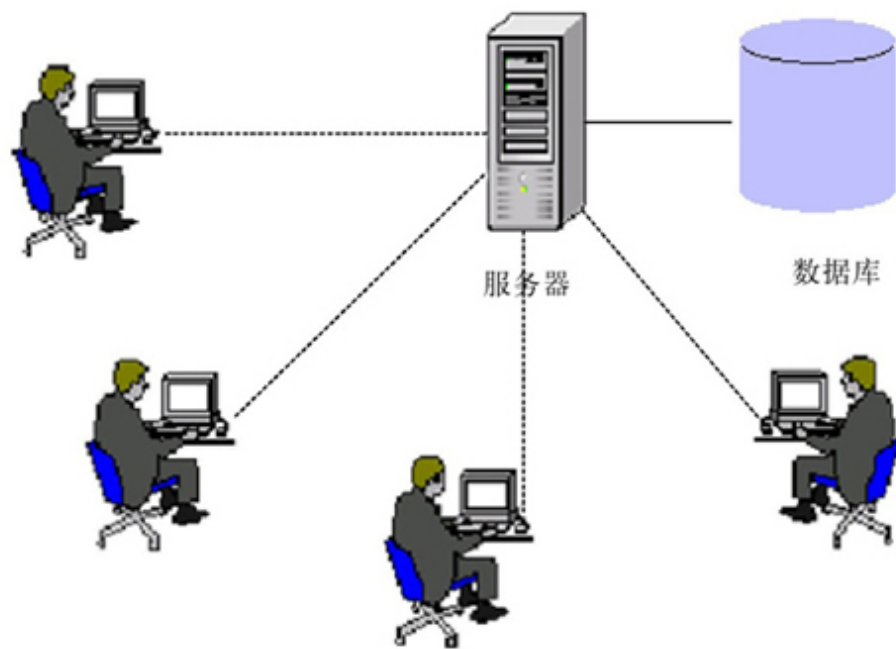


图 2-2 数据库资源共享示例

2.1.2 计算机网络的发展历史

计算机网络发展至今已经历了几代变革，不仅计算机网络的内涵发生了巨大的改变，计算机网络技术和网络应用都不再是几十年前第一、第二代计算机网络所能比拟的了。了解计算机网络的整个发展历史，有助于我们对计算机网络技术和应用的发展有一个清晰的认识，也有助于我们分清目前哪些才是主流应用的技术，哪些是已过时、我们不用去学的技术。当然要先有了计算机，才能有计算机网络，就像肯定是先有人，然后才会有人类社会一样。总体来说，可以把计算机网络的发展历程归纳为以下几个阶段。

(1) 第一代计算机网络（面向终端的计算机网络）

1946年，世界上第一台数字计算机问世。但当时的计算机数量非常少，且非常昂贵。由于当时的计算机大都采用批处理方式，所以用户计算机首先要将程序和数据打印成纸带或卡片，再送到计算中心去处理。1954年，出现了一种称为收发器（Transceiver）的终端，人们使用这种终端首次实现了将穿孔卡片上的数据通过电话线路发送到远地的计算中心计算机的过程。自此以后，电传打字也可作为远程终端与计算机相连了，用户可以在远地的电传打字机上输入自己的程序，而计算中心通过计算机计算出来的结果也可以传到远地的电传打字机上并打印出来。当时的这种简单的传输系统就是计算机网络的基本原

型。当然，这些离我们有些远，现在的我们不必研究这些收发器终端及其数据传输原理。

第一代计算机网络是以计算机主机（其实相当于我们现在所说“计算机服务器”）为中心，一台或多台终端围绕计算机主机分布在各处，而计算机主机的任务是进行成批处理，用户终端则不具备数据的存储和处理能力。从某种意义上来说，这根本不能算是真正的计算机网络，因为终端并不具备独立工作的能力。所以我们现在说，计算机网络的诞生通常不是指第一代计算机网络，而是从下面将要介绍的第二代计算机网络开始算起的。这里所说的“终端”是指由一台计算机外围设备组成的简单计算机，有点类似于现在所说的“瘦客户机”，仅包括CRT显示器、键盘，没有CPU和硬盘，所以没有数据存储和处理能力。之所以网络中更多的是计算机终端，因为那时的计算机非常昂贵，为了节省成本，所以在用户端通常是使用那些不带关键部件的计算机终端。

到了20世纪50年代中后期，通过像多路复用器（MUX）、线路集中器、前端控制器等通信控制设备，计算机网络系统可以将地理上分散的多个终端通过公用电话交换网络（PSTN）集中连接到一台主机上，这就是真正意义上的第一代计算机网络，如图2-3所示。

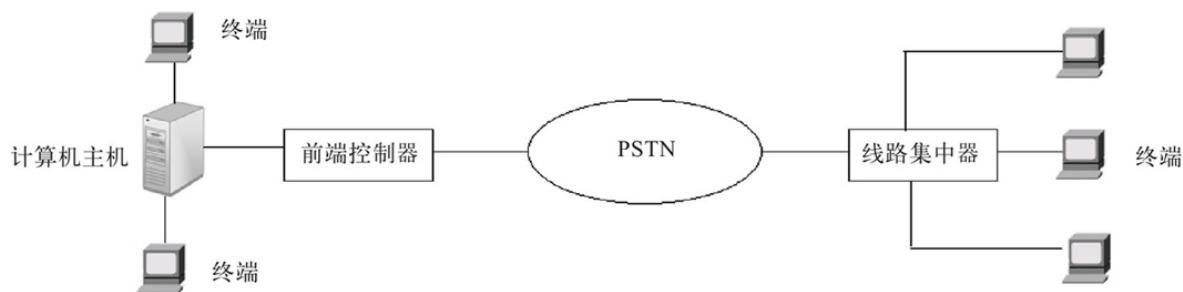


图 2-3 第一代计算机网络示例

第一代计算机网络的典型应用是当时的美国航空公司与IBM公司在20世纪50年代初开始联合研究，并于20世纪60年代投入使用的飞机订票系统SABRE-I。它由一台计算机和全美国范围内2000个终端组成。

由于第一代计算机网络是以单个计算机为中心的远程联机系统，所以很显然它具有天生的一些缺点，主要表现在如下两个方面：

□担当计算中心的计算机负荷很重，造成对终端系统的响应比较慢，甚至会出现服务器崩溃现象。

□单主机系统的可靠性较低，一旦发生计算机主机=瘫痪，将导致整个计算机网络系统瘫痪。这与现在的C/S（服务器/客户机）管理模式类似，但现在计算机网络中服务器系统通常是有容错配置的，那时的主机可没这种功能。而且现在的计算机服务器和客户机性能都非常强大，远不是当时的计算机主机和终端系统所能相提并论的。

（2）第二代计算机网络（分组交换式的计算机网络）

为了克服第一代计算机网络的缺点，提高网络的可用性和可靠性，专家们又开始研究将多台计算机互连的方法。有问题就要想办法，这与现在所有技术的改进思路是一样的。首先，1964年8月巴兰（Baran）在美国兰德（Rand）公司《论分布式通信》的研究报告中提到了“存储转发”（这在我们学习交换机技术时会介绍这项技术）的概念。在1962年至1965年间，美国的ARPA（Advanced Research Projects Agency，美国国防部高级研究计划署）和英国的NPL（National Physics Laboratory，国家物理实验室）都对这一新技术进行了研究。后来，英国NPL的戴维斯（David）于1966年首次提出了“分组”（packet）的概念。在1969年12月，产生了世界上第一个基于分组技术的计算机分组交换系统APPANET。这是大家公认的计算机网络的鼻祖。

APPANET是美国国防部高级研究计划局（DARPA）采用电话线路为主干网络建成的。它最开始仅连接了美国加州大学洛杉矶分校、加州大学圣巴巴拉分校、斯坦福大学和犹他大学四个结点的计算机；两年后建成15个结点，此后规模不断扩大。到了20世纪70年代后期，网络结点超过60个，主机100多台，地理范围跨越美洲大陆，连通了美国东部和西部的许多大学和研究机构，而且还通过通信卫星与夏威夷和欧洲地区的计算机网络相互连通。

APPANET的运行成功使计算机网络的概念发生了根本性的变化，也标志着计算机网络的发展进入了一个新的纪元。这种在计算机网络中运行各种应用程序的计算机称为主机（不再使用功能简单的终端了），这些主机提供资源共享，组成“资源子网”；各计算机之间不是直接用线路相连，而是由接口报文处理机（Interface Message Processor, IMP）转接后互连。IMP专门负责通信处理，通信线路将各IMP相互连接起来，然后各主机再与IMP相连，各主机之间的通信需要通过IMP连接起来的网络来实现。IMP和它们之间互连的通信线路一起负责主机间的通信任务，构成“通信子网”，如图2-4所示。

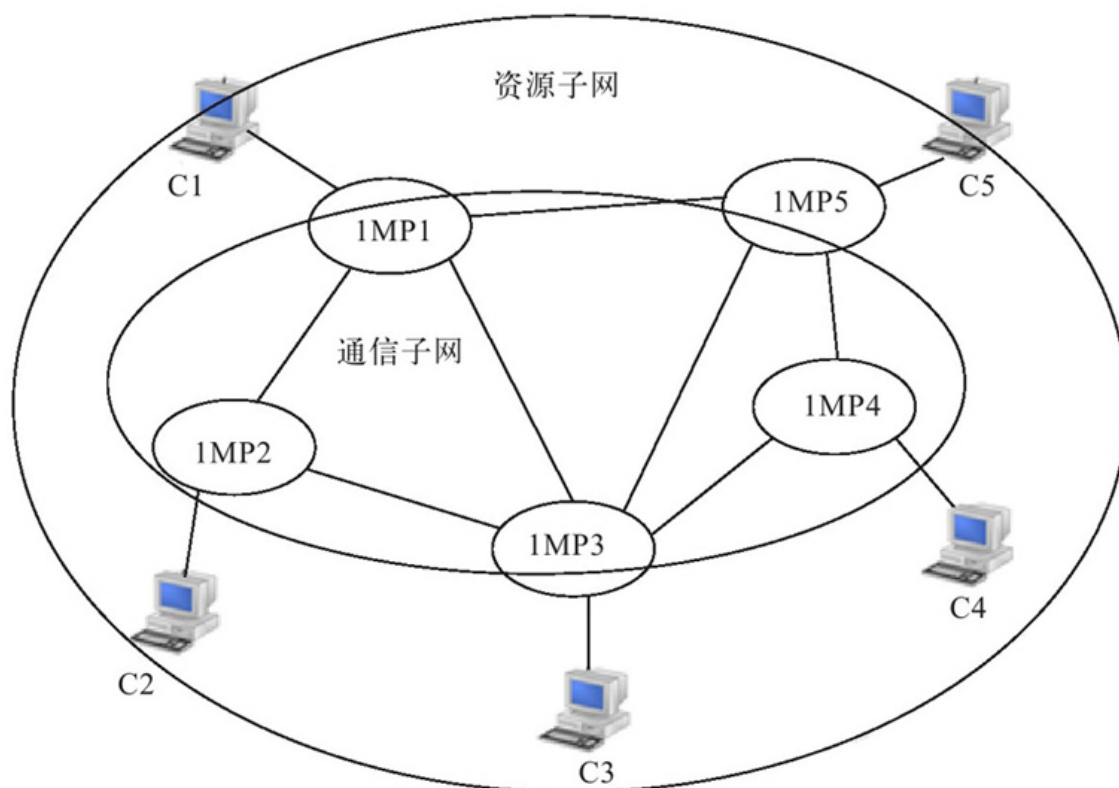


图 2-4 第二代计算机网络

第二代计算机网络中的终端用户不仅可以共享“通信子网”中的线路和设备资源，还可以共享用户“资源子网”的丰富硬件和软件资源。这种以“通信子网”为中心的计算机网络就成了我们现在所说的第二代计算机网络。

在第二代计算机网络中，采用了“存储-转发”数据通信方式，也就是各个IMP在接收到数据后先按接收顺序把数据存储在各自的缓存中，然后再按接收顺序依次进行下一级的数据转发，这样可以使网络上的流量更加平滑、有序。如果某台主机的一个用户要发送数据给网络中的另一台主机，它先是将数据交给与其相连接的IMP，接着该IMP通过适当的通信线路转给下一个IMP，然后继续转发这个数据，以此类推，直到到达目的IMP，最后由目的IMP转发给目的主机。

第二代计算机网络的这种既分散（从地理位置上来讲）又统一（从服务功能上来讲）的多主机计算机网络，使得整个计算机网络系统性能大大提高，同时也不会因为单机故障而导致整个网络系统瘫痪。另外，可以使原来第一代计算机网络中的单一计算机主机的负载可以分散到整个计算机网络的各个计算机主机上，使得计算机网络系统的响应性能也大大提高了。

（3）第三代计算机网络（标准化的计算机网络）

第二代计算机网络的传输方式采用了“存储-转发”方式，极大地提高了昂贵的通信线路资源的利用率。因为在这种“存储-转发”方式的通信过程中，通信线路不会被某一节点间的通信所独占，而是可以为多路通信所共用。但是第二代计算机网络仍存在许多弊端，主要表现为没有统一的网络体系架构和协议标准。而且，尽管第二代计算机网络已进行了通信子网和资源子网的两级分层，但不同公司的网络体系都只适用于自己公司的设备，不能进行相互连接。如IBM于1974年推出了系统网络结构（System Network Architecture，SNA），为用户提供能够互连的成套通信产品；1975年DEC公司宣布了自己的数字网络体系结构（Digital Network Architecture，DNA）；1976年UNIVAC宣布了自己的分布式通信体系结构（Distributed Communication Architecture，DCA）。这些网络技术标准只在一个公司范围内有效，遵从某种标准的、能够互连的网络通信产品，只适用于同一公司生产的设备，不同公司间的网络仍不能相互连通。计算机网络通信市场这种各自为政的状况使得用户在投资方向上无所适从，也不利于多厂商之间的公平竞争。

针对上述情况，1977年ISO（国际标准化组织）的TC97信息处理系统技术委员会SC16分技术委员会开始着手制定开放系统互连参考模型（OSI/RM），并于1984年发布。OSI/RM模型是一个开放体系结构，定义了网络互连的七层结构，并详细规定了每一层的功能（如图2-5所示），以实现开放系统环境中的互连性、互操作性和应用的可移植

性。OSI/RM模型同时规定了计算机之间只能在对应层之间进行通信，大大简化了网络通信原理，是公认的新一代计算机网络体系结构的基础，为普及计算机网络奠定了基础。



图 2-5 OSI/RM的七层结构及各层对应的基本功能

从前面第一代、第二代计算机网络技术的发展可以看出，它们都是企业驱动的，也就是由各个公司根据自己的市场和用户需求进行相关技术和产品开发的，可以说是处于“百家争鸣”的时代。虽说“百家争鸣”可以充分展示各家公司的优势，但毕竟“无规矩难成方圆”。由于这些公司间彼此的市场竞争关系，导致这些不同公司开发出来的技术和

产品并不具有通用性，这样一来结果就是谁都做不大、做不强，用户也很难选择，计算机网络的发展也就很难真正有实质性的进展。

到了第三代计算机网络时代，随着OSI/RM的诞生，就把整个计算机网络体系架构以标准形式确定下来了，大大推动了计算机网络的发展，激发了无数公司参与开发计算机网络相关硬件和软件，迎来了计算机网络发展历史上第一个真正意义上的“春天”。因为大家已“有规可循”了，不再担心兼容问题了。当然OSI/RM标准的诞生也是在汇总了前面不同公司开发的体系架构优点的基础上开发的。

OSI/RM的诞生也标志着第三代计算机网络的诞生。此时的计算机网络在共同遵循OSI标准的基础上，形成了一个具有统一计算机网络体系结构，并遵循国际标准的开放式和标准化的网络。1980年2月，IEEE学会下属的802局域网标准委员会宣告成立，并相继推出了若干个802局域网协议标准，其中绝大部分后来被OSI正式认可，并成为局域网的国际标准。这标志着局域网协议及标准化工作向前迈出了一大步。从1980年至今，802委员会已陆续发布了环网、总线网、令牌总线网、光纤网、宽带网、城域网和无线局域网等许多局域网标准。

这些IEEE 802局域网标准的制定，极大地推进了计算机局域网的发展。近几年，计算机局域网的发展速度更是惊人，千兆、万兆、十万兆的局域网技术也已走入应用，百万兆的局域网技术的研究也取得了非常不错的进展。不仅如此，以太网的传输距离已从原来局域网的

范围延伸到了城域网、广域网的范围。正因如此，现在的局域网和广域网之间的界限变得越来越模糊了。事实上，现在，几乎每个单位的计算机网络都接入了Internet，每个人的计算机也都已接入了Internet，成为了Internet的一分子。

最后，不得不说的是，虽然OSI/RM的诞生大大促进了计算机网络的发展，但主要还是表现在局域网范围中，在后来的广域网，包括Internet（互联网）的发展中，OSI/RM却被后来居上的TCP/IP协议规范（由DARPA研究并发布）远远抛在后面。1983年，DARPA将ARPANET上的所有计算机结构转向了TCP/IP协议，并以ARPANET为主干建立和发展了Internet，形成了TCP/IP体系结构。

TCP/IP协议体系结构虽然不是国际标准，但它的发展和应用都远远超过了OSI/RM，成为了Internet体系结构上的实际标准。究其原因主要有以下三个方面：一是TCP/IP协议簇非常庞大，功能完善且实用（目前的Internet基本上全是TCP/IP协议类型的网络），用户基础好；二是曾经的Internet的投资者不会轻易放弃在TCP/IP协议体系上的巨大投资；三是OSI/RM的网络体系结构本身分层过多，有些层次（如会话层和表示层）没有太大单独划分的必要性，而有些功能（如流量控制和差错控制等）又在多个层次中出现，实现和协调起来比较难。

当然，我们不能否认OSIR/RM的贡献，它提出了许多计算机网络的概念和技术至今仍广为使用，包括在Internet上。另外，也正是在它

的推动下，使得计算机网络体系结构的标准化工作不断进展，事实上后来的TCP/IP协议规范也是在OSI/RM基础上改进而来的。

说明 有关计算机网络体系结构方面的具体内容将在第3章具体介绍，有关这些计算机网络体系结构中各层的主要功能和技术实现原理将在后续各章中分别介绍。

（4）第四代计算机网络（国际化的计算机网络）

到了20世纪80年代末，局域网技术发展成熟，出现了光纤及高速以太网技术，局域网的组建和应用也首先在国外，特别是美国开始普及开来。随着第三代计算机网络中的OSI/RM体系架构的诞生，又大大促进了以Internet为代表的因特网的发展，这就是现在的第四代计算机网络。第四代计算机网络的定义为“将多个具有独立工作能力的计算机系统通过通信设备和线路由功能完善的网络软件实现资源共享和数据通信的系统”。其实，Internet的雏形就是DARPA的ARPANET，所采用的协议标准就是TCP/IP协议规范。

Internet的基本发展历史如下：1985年美国国家科学基金会（National Science Foundation）利用ARPANET协议建立了用于科学研究和教育的骨干网络NSFnet；1990年NSFnet取代ARPANET成为国家骨干网，并且走出了大学和研究机构进入社会，从此网上的电子邮件、文件下载和信息传输受到人们的欢迎和广泛使用；1992年，Internet学会

成立；1993年，伊利诺斯大学国家超级计算中心成功开发出网上浏览工具Mosaic（后来发展为Netscape），同年克林顿宣布正式实施国家信息基础设施（National Information Infrastructure）计划，从此在世界范围内开始了争夺信息化社会领导权和制高点的竞争；与此同时NSF不再向Internet注入资金，完全使其进入商业化运作；20世纪90年代后期，Internet以惊人的速度发展，一直到今天。

（5）下一代计算机网络

有人会问，我们现在的计算机处于一个什么时代？可以这么说，我们目前正处于第四代和第五代之间的过渡时期。但最终下一代网络到底是什么样子，现在可能还没有人能全部说清楚，至少没有形成标准。总体而言，普遍认为下一代计算机网络（NGN，也就是我们所说的第五代计算机网络）是因特网、移动通信网络、固定电话通信网络的融合，IP网络和光网络的融合；是可以提供包括语音、数据和多媒体等各种业务的综合开放的网络构架；是业务驱动、业务与呼叫控制分离、呼叫与承载分离的网络；是基于统一协议的、基于分组的网络。在功能上NGN分为四层，即接入和传输层、媒体层、控制层、网络服务层。我们看得见的一些下一代计算机网络的主要特征包括：目前正在进行的“三网”（计算机网络、电信网络、广播电视网络）融合，物联网、虚拟化、云计算、HTML5等新的革命性技术等。

在这些新技术中，“云计算”和“物联网”可能是将来彻底改变目前计算机网络格局和应用的最主要的两大技术。“云计算”其实类似于以前的IBM大型机，是一种集中服务、集中管理的平台。就是由云计算运营商集中为企业客户提供一些软、硬件平台及各种所需的服务和管理，企业客户只需要通过比较简单的云计算客户端连上位于互联网上的运营商云计算平台就可享受所购买的服务、平台，大大节省了企业客户在计算机网络软、硬件平台（如各种服务器系统、企业交换机、路由器、防火墙等）上的投资。

“物联网”是一种继续扩展计算机网络的新型技术，简单地讲就是“物-物相联的网”。它是通过射频识别（RFID）技术，以及红外感应器、全球定位系统、激光扫描器等信息传感设备，按约定的协议，把一些目前不能与计算机网络连接的物品（如电灯、电器、监控设施等）与互联网连接，以便进行物品之间的信息交换和通信，实现对物品的智能化识别、定位、跟踪、监控和管理。通过“物联网”我们将来在上班时就可以控制家里的电灯、电器设施的开关，监控家里的防盗监控设施，真正实现无处不在的互联网应用。

2.1.3 计算机网络的基本组成

无论是从前面哪个定义都可以看出，计算机网络是一个由一些硬件设备和相应的软件系统组成的完整系统。计算机网络的基本组成包括：计算机（或者是只具有基本计算机功能的计算机终端）、网络连
接和通信设备、传输介质、网络通信软件（包括网络通信协议）。

以上这些计算机网络基本组成又分为硬件系统和软件系统两大部分，如图2-6所示。

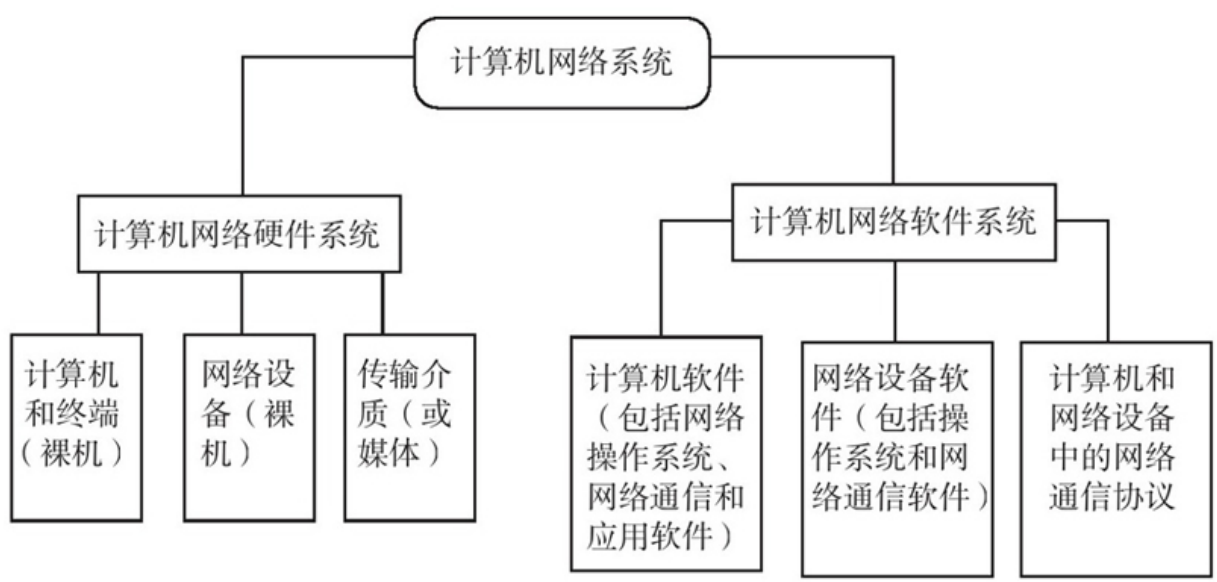


图 2-6 计算机网络系统组成

1. 计算机网络硬件系统

计算机网络硬件系统就是指计算机网络中可以看得见的物理设施，包括各种计算机设备、传输介质、网络设备这三大部分。

（1）计算机设备

组建计算机网络的目的是为各种计算机设备用户之间的网络通信（可以是用户访问、数据传输、文件共享、远程控制等应用）提供平台。计算机设备就是由网络用户控制和使用的各种计算机（如PC、计算机服务器、计算机终端、笔记本式计算机、iPAD之类的便携式设备）。网络的主要应用都是在这些计算机设备上进行的。其实现在计算机网络与电信通信网络有些重合了，许多电信通信终端同样可以连接到计算机网络中，如我们现在所使用的智能手机，就可以通过USB接口与计算机之间进行数据传输，甚至进行远程通信。

说明 在传统的计算机网络定义中，计算机网络至少要求有一台功能完整的物理计算机（其他的可以是终端）。随着网络虚拟化技术的兴起，目前的计算机网络可以通过虚拟机软件（如VPC、VMWare等）在一台物理计算机中模拟多个独立计算机系统，组成一个虚拟的计算机网络，通过这个网络同样可以实现在许多物理计算机网络中才能实现的功能。

（2）网络设备

在计算机网络系统中，网络设备通常是指除计算机设备以外的设备，如有线网络中的网卡、网桥、网关、Modem、交换机、路由器、硬件防火墙、硬件IDS（入侵检测系统）、硬件IPS（入侵防御系统）、宽带接入服务器（BRAS）、UPS（不间断电源）等，无线网络（WLAN）中的WLAN网卡、WLAN AP、WLAN路由器、WLAN交换机等。有关这些计算机网络设备的主要用途和特点将在本书后面专门介绍。

网络设备是用来构建“通信子网”中网络拓扑结构的，与所用的通信线路（也就是“传输介质”）一起共同组成整个计算机网络的骨架。当然最简单的网络，其实是不需任何网络设备的，那就是两台终端计算机用串/并口电缆直接连接起来的对等网络。但这种网络其实并不能算是真正意义上的计算机网络，对于现在来说，这样的计算机网络也没多大的实际意义。

（3）传输介质

传输介质简单地说就是网线，是网络通信的“路”。如果没有这些传输介质，网络通信信号将不知道往哪里传，也不可能传，就像前面没有路，我们无法向前行进一样。当然，传输介质可以是物理有形的，如同轴电缆（有线电视所用的电缆也是同轴电缆）、双绞线、光缆（也常叫光纤）等（依次如图2-7的a、b、c所示）；还可以是无形的，如各种无线网络中使用的传输介质其实就是电磁波。无线计算机

网络就是通过电磁波实现无线计算机网络中各节点连接的。当然，在同轴电缆、双绞线、光缆这些传输介质中又有许多分类，具体将在第4章介绍。

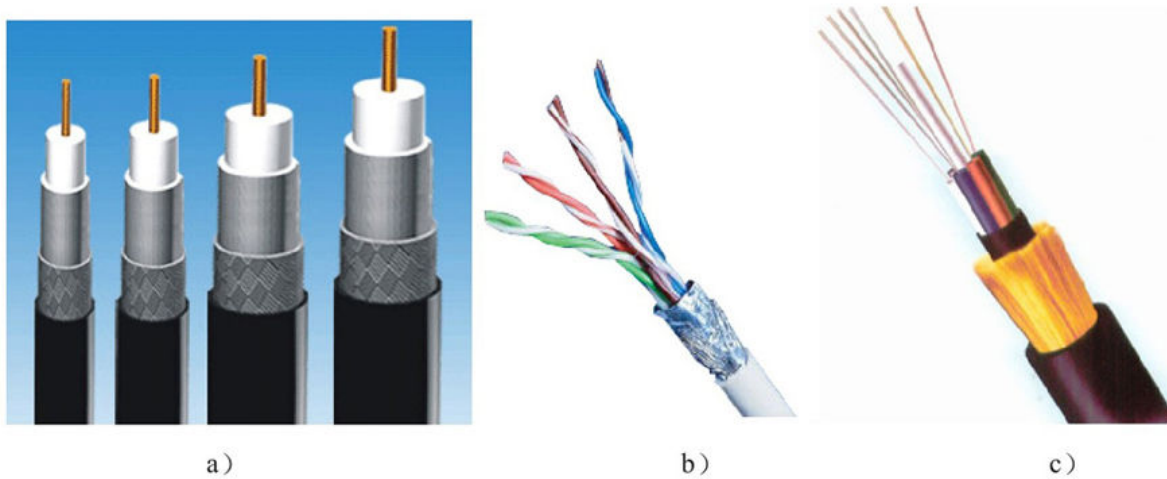


图 2-7 同轴电缆、双绞线、光缆

2.计算机网络软件系统

计算机网络通信除了需要前面所说的各种计算机硬件系统外，还需要一些计算机网络通信和应用软件。这些计算机网络通信和应用软件就是指安装在终端计算机中，用于计算机网络通信或应用的计算机程序。首先要有的就是一个网络应用平台，如计算机和服务服务器上所安装的、具备计算机网络通信功能的操作系统。像交换机、路由器和防火墙等，这类设备上也会安装用于计算机网络通信的操作系统。如计算机或服务服务器上所安装的各种Windows系统、Linux系统、UNIX系统，

Cisco交换机/路由器/防火墙上安装的CatOS、IOS系统，H3C交换机/路由器/防火墙上安装的Comware系统等。

除了操作系统以外，还需要独立或者内植于操作系统中的网络通信协议，如TCP/IP协议簇、IEEE 802协议簇、PPP、PPPoE、IPX/SPX等，及网络设备中的VLAN、STP、RIP、OSPF、BGP等。最后就是需要进行各种具体网络应用的工具软件，如我们常见的QQ、MSN等即时通信软件，Outlook、Firefox、Sendmail等电子邮件软件，用于拨号的PPP、PPPoE协议，用于VPN通信的IPSec、PPTP、L2TP协议等。

说明 有关网络设备中的各种通信协议的具体工作原理和应用配置，可参见笔者编著的《Cisco交换机配置与管理完全手册》、《H3C交换机配置与管理完全手册》、《Cisco路由器配置与管理完全手册》和《H3C路由器配置与管理完全手册》这四本专著（这四本图书将于2013年6月份前出版上市）。

2.1.4 计算机网络的主要应用

说到计算机网络，大家肯定首先要问的是它到底有什么用途，也就是我们可以用它来做什么。如果时光倒退十多年，问到计算机网络有什么用途时，可能大家会不约而同地回答为“资源共享”。那时的计算机网络应用的确如此，除了主要用于资源共享外，看不出有其他方面的应用，因为那时既没有互联网，又没有什么局域网内部的网络应用。现在，随着计算机网络系统，特别是Internet技术的完善与普及，计算机网络的应用得到了空前繁荣。已渗透到了普通百姓的日常工作、生活和休闲等各个方面。可以说，我们现在是生活在网络时代，关于计算机网络的用途，绝大多数普通计算机网络用户都可以说出个一、二、三来。

虽然目前的计算机网络应用非常多，但总体上可以分为两大类：商业应用、家庭/个人应用。下面进行具体介绍，大家可以对照地看一下自己公司，乃至本人用到了哪些。

1.商业应用

商业应用是计算机网络应用的最主要的方面，后面的家庭/个人应用是在商业应用基础上发展而来的。在计算机网络的商业应用中，主要依赖的网络就是公司局域网，以及与公司局域网连接的外部用户

（如子公司、合作伙伴、供应商等）的内部局域网和Internet。商业应用主要包括：资源共享、数据传输、协同工作、远程访问与管理、电子商务等。

（1）资源共享

在计算机网络中，一个最基本，也是最传统的应用就是资源共享（Resource Sharing）。这里的共享资源可以是物理设备，如打印机、扫描仪、传真机、刻录机，也可以是共享的数据文件、软件资源等。其目标就是让每个人都可以访问其允许的设备、程序、文件和数据信息。一个最简单的资源共享的例子就是局域网内部多用户通过网络共享一台打印机（现在有些打印机也支持Internet共享打印）进行打印操作，这样一来，公司就没有必要为每个用户配备一台单独的打印机了，大大节省了设备投资成本。

在公司局域网中，比物理设备共享意义更重要的还是我们前面提到的程序、文件和数据资源共享，如公司内部公用文档、数据库报表，或者用户共享安装使用的软件，以及公司数据库系统。局域网的好处就是我们无需再像没有计算机网络时那样，需要通过移动媒体（如软盘、U盘、移动硬盘等）来复制数据了。这样既确保了共享数据的安全性（因为共享时还可以为不同用户设置不同的访问权限），又大大提高了数据共享使用的效率。通常在公司内部局域网中，会有一个文件服务器来存储这些共享数据资源。

在互联网中的资源共享实例就更多了，如许多网站提供的文件上传和下载功能，音、视频分享功能，文件浏览和查阅功能等。

（2）网络通信

在网络通信方面，目前在企业中用得较多的功能是远程网络互联、远程视频会议，远程培训、远程会诊等。远程网络互联目前在集团公司与子公司之间，或者公司与合作伙伴、供应商网站之间通过专门的接入方式（目前主要是利用VPN技术）把各单位的网络按照应用需求和访问权限连接起来。这样可以使网络间的通信、用户访问更加安全、便利，还可以更有效地管理网站数据、电子商务数据等。

远程视频会议、远程培训、远程会诊等的应用目前在一些大的集团公司或医院中是很常见的。应用这个功能一方面可以节省会议、培训成本（因为它克服了物理距离的时空限制），另一方面可以充分利用各方面的专家资源，及时地解决一些疑难杂症。

（3）数据传输

在计算机网络中进行数据传输是最常见的了，如我们天天在用E-mail邮件收发、通过FTP（文件传输协议）进行文件传输、通过TFTP/RCP（简单文件传输协议/远程复制协议）进行文件上传和下载等。又如我们可能天天都在通过QQ或者MSN等工具软件向好友发送文件；许多网站提供了资源下载功能，供用户根据需要选择下载他们

资源库中的文件；现在又有了一些专门用于资源上传和下载的网站网盘等。这些上传和下载的过程都属于计算机网络的数据传输应用。

（4）协同工作

协同工作是目前计算机网络的一种典型应用，是指通过网络，使位于相同或者不同地点，甚至不同国家的多个系统共同担负着某项网络通信或者网络应用任务的工作方式。最典型的例子如服务器、交换机集群实现的负载均衡，ISP中的多个DNS服务器、DC（域控制器）服务器等也可以实现负载均衡，为网络用户提供相应的服务。

协同工作目前还有一个应用就是现在的维基（Wiki）百科，其中的内容可以由全球授权参与的网友共同编辑、完善。还有，如一个项目可以由总公司和分公司的多人共同负责完成。

（5）远程访问与管理

远程访问与管理是计算机网络用户访问，以及管理员对客户机和服务器的管理方式。如一些支持移动上网的VPN解决方案就可以使公司员工在任何时候、任何地点通过VPN连接到公司的网络，查看所需的文件或数据，上传或下载所需的文件。另外，利用像Windows服务器系统中的“远程Web桌面”和“远程协助”功能就可以使在外出差的员工通过Internet访问甚至控制公司内部网络中的主机或服务器。如果具有管理员账户，他们还可以远程管理、维护公司内部的服务器。

（6）电子商务

现在几乎所有稍具规模的单位都架设了自己的网站，目的之一就是向全世界的用户宣传自己的产品。另外还有大部分企业用户通过自己的网站为客户提供网上在线交易，这就是我们通常所说的“电子商务”。相信大多数人都有过在淘宝网购物的体验，淘宝网上有许多商家开店销售他们的商品，这些商品大到名贵珠宝、家用电器，小到日用百货和图书，一应俱全。还有像当当网、亚马逊卓越网、京东商城等（目前这类网店非常多，数不胜数了）都允许各种商品在网站上销售。现在通常把这些电子商务网站称为“电商”，以区别于实体开店的商户。

“电子商务”这种经营模式最大的优点有三个：一是受众面广，全国乃至全球的人都可以看到；二是成本低，因为这类电子商务网站不需要租用昂贵的实体店面，只需要利用虚拟的磁盘空间就可以了；三是用户购买方便，只需要在网上提交所购商品订单，就可以坐在家等着收货了，而不用亲自跑到商店中去买，然后还要想办法运回来。

2.家庭应用

一开始，计算机网络基本上全部是出于商业应用的，但随着Internet宽带接入和互联网应用的丰富，计算机网络开始走入寻常百姓的家。现在，我们坐在家就可以通过家里的宽带接入访问互联网，

访问全球的网站；通过像QQ、MSN之类的即时通信软件与世界各地的相识或不相识的朋友取得即时联系，建立感情；我们还可以有自己的局域网，或者个人网站，全世界的人都可以访问我们的网站，了解我们提供的产品、服务，以及我们的工作、学习和生活状态。

在娱乐方面，现在各大电视台都开通了网上频道，各种娱乐节目可以直接通过互联网进行互动参与。网上看电影、听音乐和手机上网娱乐更是年青一族的时尚追求（现在电信推出的IPTV已实现了电影、电视、音乐、游戏等各方面的互动点播，就像在自己的电脑中播放一样）；而网络游戏则成了年青一族的最爱（还记得你与朋友们在网络游戏中一起拼杀的场景吗？）。

当然，现在的计算机网络应用可能还远不止以上这些，因为它已渗透到了我们工作、生活、学习和娱乐等各个方面，而且各种新的网络应用也在不断涌现中。

2.2 计算机网络的分类

自计算机网络诞生至今，出现过许多类型的计算机网络，可以依据许多不同的分类标准来划分这些计算机网络。如按通信协议类型分为IBM令牌网络、分组交换网络、以太网协议局域网、TCP/IP协议网络等；按管理模式则可分为对等网和C/S（客户机/服务器）网络；按不同传输介质则可分为同轴电缆网络、双绞线网络、光纤网络、WLAN无线网络、卫星网络、微波网络等；按传输方式来分又可分为点对点网络和广播网络；按所覆盖的地理范围又可分为局域网、城域网和广域网；等等。下面介绍几种目前最常见的计算机网络分类方式。

2.2.1 按网络所覆盖的地理范围分

按照计算机网络覆盖的地理范围（也是一种按网络规模进行的计算机网络分类）可分为局域网、城域网和广域网三种。这种分类方式可以很好地反映出不同类型网络的技术特征，因为不同类型网络覆盖的地理范围不同，故所采用的传输技术也就不同，从而形成了不同的网络技术特点与网络服务功能。

1.局域网

局域网（**Local Area Network, LAN**）是最常见到的，也是应用最多的一种计算机网络，大到各行各业的企业内部网络，小到千家万户的家庭网络都属于局域网（仅指内部网络部分）。我们常说的校园网通常也是一种局域网。局域网是将一个比较小的区域内的各种通信设备互连在一起组成的计算机网络。**LAN**具有如下主要特点：

（1）私有服务

LAN属于个人或单位自建，所以其用途也是完全出于私用，不会为网络以外人员提供服务，如企业局域网通常只为本单位员工提供服务。在**LAN**中也是采用专门为**LAN**分配的私有**IP**地址（这种地址每个公司和个人都可直接拿来使用，无须购买，也无须注册）。关于局域网的具体内容将在第8章介绍。

（2）分布范围较小

LAN中各计算机网络设备分布的地理范围较小，有的甚至只是在自己家里那几平方米范围内，当然也可以是较大范围内的设备相连，如分布在某公司不同建筑物中。**LAN**的地理分布范围通常最大在10公里范围内，可以分布在不同地理位置的建筑物内。

（3）结构简单，布线容易

因为LAN都为个人或单位私用，所以网络结构相对较为简单。没有，也无需太多、太复杂的网络设备和应用，只需满足自身的网络应用需求即可。同时大多数LAN都采用比较廉价的双绞线布线（在较大公司的核心层或者汇聚层也有采用光纤作为传输介质的），且因为分布范围比较小，所以布线方式较为简单，容易实现。

（4）网络速度较快

目前以太网局域网技术的发展非常迅速，最快的以太网速率已达到了10Gbps，相对广域网和互联网来说具有非常大的优势，这也为企业局域网的集中应用提供了保证。

（5）误码率低

因为局域网的结构比较简单，而网络连接带宽又都很高，所以通信的误码率比较低，通常在 $10^{-11} \sim 10^{-8}$ 之间。

在LAN中又有许多分类，如目前最主流的以太网（EtherNet）、WLAN（无线局域网），以前还有IBM令牌网（传输速率最高可达16Mbps）、FDDI（光纤分布式数据接口）网（最高传输速率可达100Mbps）、ATM（异步传输模式）网（最高传输速率可达Gbps以上）。而以太网中又有许多分类，如10Mbps的标准以太网、100Mbps的快速以太网、1Gbps的千兆以太网、10Gbps的万兆以太网，现在

100Gbps的十万兆以太网也正在研究之中。有关以太网标准的内容将在第6章介绍。

2.城域网

城域网（Metropolitan Area Network, MAN）中各计算机网络设备的地理分布范围介于LAN和下面将要介绍的广域网（WAN）之间，主要遍布一个城市内部，所以称之为“城域网”。MAN主要是用来在一个较大的地理区域（通常是10~100公里）内提供数据、声音和图像的传输，一般是用于提供公共服务的。

MAN的标准为IEEE 802.6，通常采用ATM技术作为骨干网传输技术，目前光纤技术也在城域网中得到了广泛应用。ATM是一个可同时应用于数据、语音、视频和其他多媒体应用的网络传输技术。城域网通常为一个或几个组织所有，更多的是为公众提供公共服务的，如城市银行系统、城市消防系统、城市邮政系统、城市有线电视/广播网络等。

3.广域网

广域网（Wide Area Network, WAN）是规模最大的一种计算机网络，分布的地理范围可以非常广，如一个或多个城市，或者多个国家，甚至可以遍布全球。Internet是最大的广域网。它遍及全球，由全球许多LAN、MAN互联组成。WAN主要也是为公众提供公共服务的，

由不同ISP（Internet服务商）组建，为他们的广大用户提供各种网络接入和应用服务。WAN具有以下基本特性。

（1）覆盖范围广

WAN所覆盖的地理范围非常大，一般从几百公里到几千公里，可覆盖多个城市、整个国家，乃至全球。Internet是最大的广域网，其他的WAN又是Internet的核心。在广域网中通常使用的是公网IP地址（这种地址是需要购买并注册的）。当然，WAN互连的各局域网内部仍可使用供局域网私用的IP地址。

（2）构建成本高

由于WAN地理范围广，网络线路很长，介质类型多种多样，而且铺设非常困难，所以单独组建一个WAN的成本非常昂贵，所以通常借用传统的公共传输（电报、电话）网这个平台来实现。

（3）网络结构和类型复杂

由于WAN连接了多个远程网络，所以网络结构非常复杂、网络类型也可能不一样，所以需要解决不同结构和不同类型网络之间的互连的问题。

（4）传输速率低，误码率高

由于WAN的传输距离远，又依靠传统的公共传输网，所以误码率较高（一般在 $10^{-8} \sim 10^{-7}$ 之间），传输速率较低（通常是100Mbps以内）。

图2-8显示了局域网、城域网和广域网三者之间的典型关系。其中的“接入网”是位于局域网与城域网或广域网之间的中间小型网络，专门为远程网络间互连提供网络用户接入技术，如各种ISP（Internet服务商）为我们提供的ADSL拨号、光纤接入、分组接入、卫星接入，以及各种专线接入等。

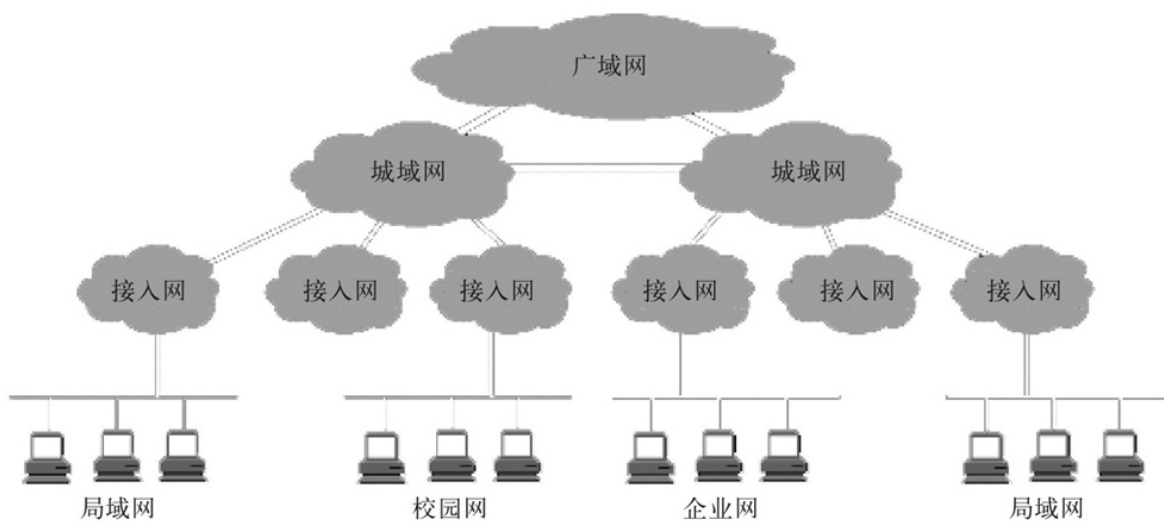


图 2-8 局域网、城域网和广域网之间的关系

2.2.2 按网络管理模式分

按计算机网络的管理模式可以把目前的计算机网络划分为对等网（Peer-to-Peer, PTP）和C/S（Client/Server, 客户机/服务器）网。

1.对等网

所谓“对等网”（PTP），即网络中各成员计算机的地位都是平等的，没有管理与被管理之分。计算机各自为政，谁也不管谁，采用的是分散管理模式。就像日常的一些小沙龙一样，沙龙中各成员是平等的，大家组织在一起仅为了相互交流。对等网中的每台计算机都既可以作为其他计算机资源访问的服务器，又可作为工作站来访问其他计算机，整个网络中没有专门的资源服务器，如图2-9所示。最简单的对等网可以仅通过串行线缆（称为零调制解调器）来连接两台计算机。

对等网可以说是当今最简单的网络，远没有像Windows域网络那样的C/S网络配置复杂，非常适合家庭、校园和小型办公室用户。从用户和计算机管理角度来看，通过Linux和UNIX操作系统组成的计算机网络都采用这种网络管理模式，Windows操作系统中的“工作组”网络也是对等网管理模式。但要注意的是，即使在对等网中，也可能有部分服务是采用C/S管理模式的，如在工作组网络中部署的文件服务器、数据库服务器、邮件服务器等。

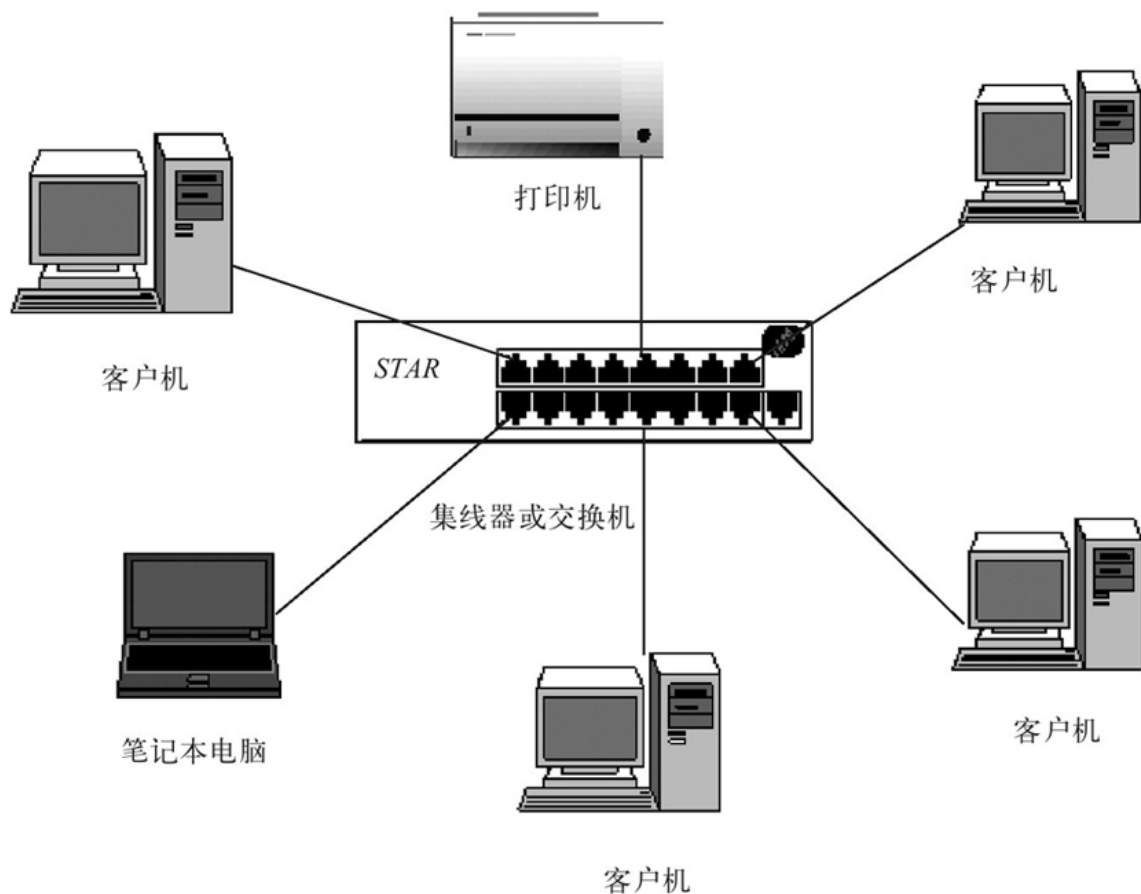


图 2-9 对等网示例

对等网除具有配置简单的优点之外，更多的是不足，主要体现在以下几个方面：

(1) 配置与管理困难

因为采用的是分散管理模式，所以对等网络管理员无法做到通过一台计算机集中管理网络中的用户、计算机和其他资源，更难以对网络中的用户计算机进行统一配置。

(2) 安全性差

同样是因为对等网采用分散管理模式，给整个网络（特别是企业网络）的安全性带来了巨大挑战。因为在这种计算机网络中，企业数据分散保存在各用户计算机上，很难十分有效地为每一用户计算机配置高级的安全保护措施。

(3) 成本高

表面上看对等网不需要服务器，成本会更低，事实上不完全是这样，特别是对于有一定规模的企业网络。因为企业数据是分散在各用户计算机上存储和管理的，随着时间的推移，可能需要为每个用户计算机配备较大容量的磁盘，浪费了磁盘空间。如果要再为每个用户配备相应的安全保护措施，这成本就更高了。

(4) 性能差

在对等网中，各用户计算机的网络连接性能、数据处理性能和磁盘读取性能基本上一致，而且通常只是普通的性能，再加上各用户计算机之间可能因一些数据共享而需频繁访问，所以对等网的网络通信和数据读取性能就只能算一般了。而在下面将要介绍的C/S模式网络中，服务器的网络连接性能、数据处理性能和磁盘读取性（通常是采用服务器专用磁盘）能都比较高，而且用户间的访问通常比较少，所以C/S网络的整体网络性能往往会更高。

2.C/S网

C/S模式其实是针对具体服务器功能来说的，这些服务器可以是用于管理整个计算机网络中计算机和用户账户的服务器（如Windows域网络中的域控制器），也可以是其他网络或应用服务器（如邮件服务器、数据库服务器、Web服务器、FTP服务器等）。这些服务器有一个共同的特点，就是一般只作为服务器角色而存在，专门为网络中其他用户计算机提供对应的服务。这相当于我们现实生活中每个组织、每个部门都有一个或多个管理者一样，他们负责的就是对对应的组织或部门进行管理。图2-10所示的就是一个单台服务器的C/S网，而图2-11所示的是一个具有多台不同类型服务器的C/S网。

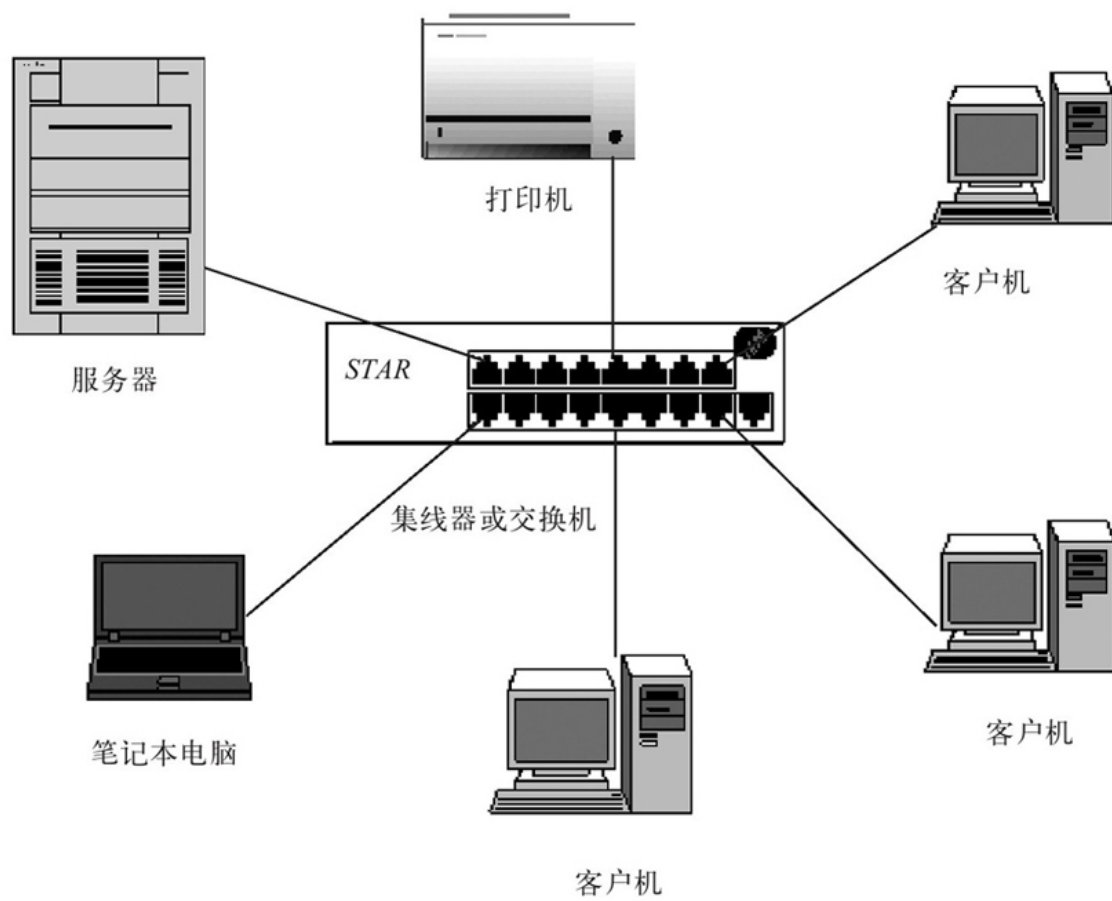


图 2-10 单服务器的C/S网

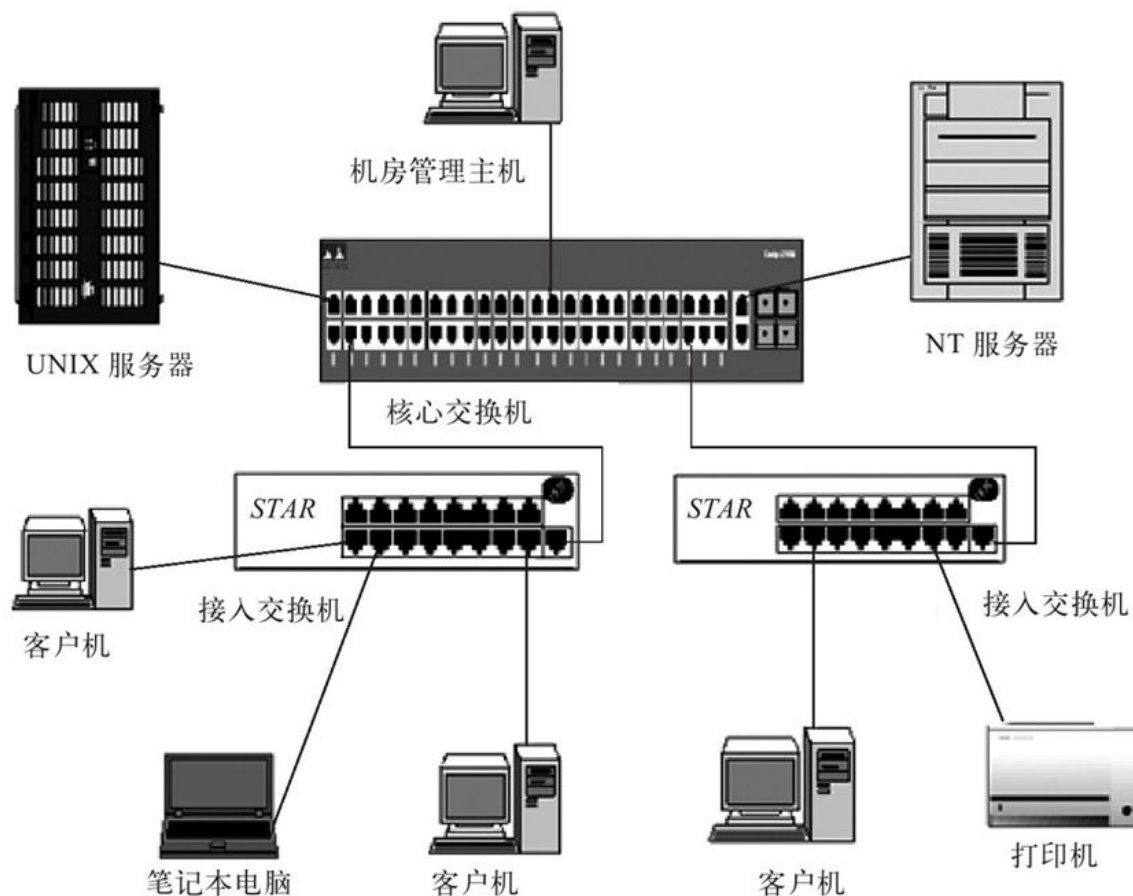


图 2-11 多服务器的C/S网

C/S网的杰出代表就是Windows服务器系统（如Windows Server 2003、Windows Server 2008等）的域网络。这种域网络可以对网络中的所有用户、计算机账户进行统一管理。除此之外，各种其他网络服务器（如DNS服务器、DHCP服务器、NFS服务器等）和应用服务器（如Web服务器、FTP服务器、E-mail服务器等）也可以组成C/S网，不过它们不能集中管理网络中的用户和计算机账户。

综合起来，C/S网络的主要优点如下：

（1）管理和配置容易

在C/S网中，用户账户往往可以集中管理，在服务器上的配置可能稍微复杂一些，但在客户端的配置却非常简单；对于Windows域网络，更是可以集中管理整个网络中的用户和计算机账户。

（2）安全性高

同样是因为在服务器上集中管理了用户账户、计算机账户、企业数据，乃至整个网络的安全策略，所以整个网络的安全性较高。

（3）性能好

因为网络中有专门的高性能服务器，同时分配有高性能的交换处理设备、高带宽端口和高性能磁盘（通常还使用性能和安全性更高的磁盘阵列），所以整个网络中尽管用户会集中频繁访问服务器，但就整体网络性能来讲，仍比对等网模式要高。

C/S网对于小型企业来说，因为要专门配备高性能的服务器，所以成本还是有些高。但对于稍有一些规模的企业网络来说，这点成本其实完全可以从各用户计算机磁盘容量成本中节省出来，因为有了专门的服务器后，各用户计算机的磁盘容量可以小许多，自然总成本就低了许多。目前各企业基本上都采用的是C/S网，因为企业中基本上都会

专门配置一台甚至多台专门的服务器，哪怕是Linux和UNIX操作系统网络。

2.2.3 按传输方式分

按网络传输方式计算机网络可划分为点对点传输网络和广播式传输网络两种。这种划分方式其实是根据所采用的传输协议进行划分的方式，因为无论是点对点传输网络，还是广播式传输网络，都主要取决于所采用的通信协议，与网络拓扑结构也有一定的关系。

(1) 点对点传输网络

在点对点传输网络中采用的通信协议都是基于点对点通信的，如SLIP（串行线路Internet协议）、PPP（点对点协议）、PPPOE（基于以太网的点对点协议）、PPTP（点对点隧道协议）等。我们使用的各种Modem拨号（拨号网络的基本结构如图2-12所示），以及路由器间串口（通常称为S口）的连接，所使用的都是PPP（点对点协议）或PPPOE（以太网点对点协议）。我们打电话也是点对点通信的，通信只在两部电话机线路之间进行，其他线路上的用户是听不到的。

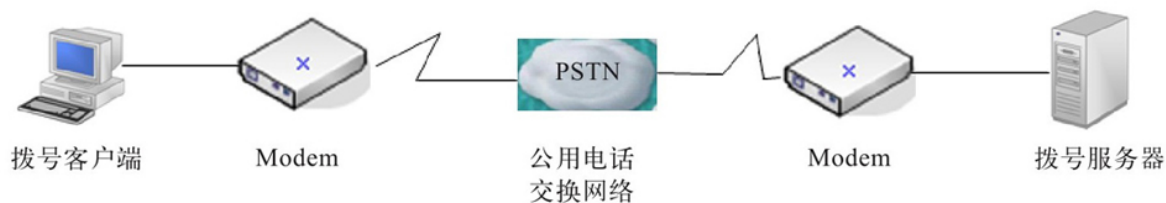


图 2-12 拨号网络基本结构

在点对点传输网络中，数据是以点对点的方式（或者说是“一对一”方式）在计算机或通信设备中传输的，也就是某个端口只能和与它相接、相连的对端端口进行通信，不能把数据发送到本网络的其他链路中，也就是只能单点“联系”。就像我们在电视剧中经常听到的“单线联系”一样，某个人只能与组织中另一个人联系，任何一个下级成员根本不知道整个组织中其他成员的联系方式。

点对点传输网络是由许多互相连接的节点构成的，在每对机器之间都有一条专用的通信信道，也就是说这两台机器是独占通信线路的，如各种拨号网络就是这样的。因此在点对点传输网络中，不存在信道共享与复用的情况。

（2）广播式传输网络

广播式传输网络是一种可以仅使用由网络上的所有节点共享的公共信道进行广播传输的计算机网络，是一种一点对多点的网络结构。图2-13就是一个广播式传输网络示例，图中PC1机发送一个广播包，可以到达网络中其他任何PC机上。

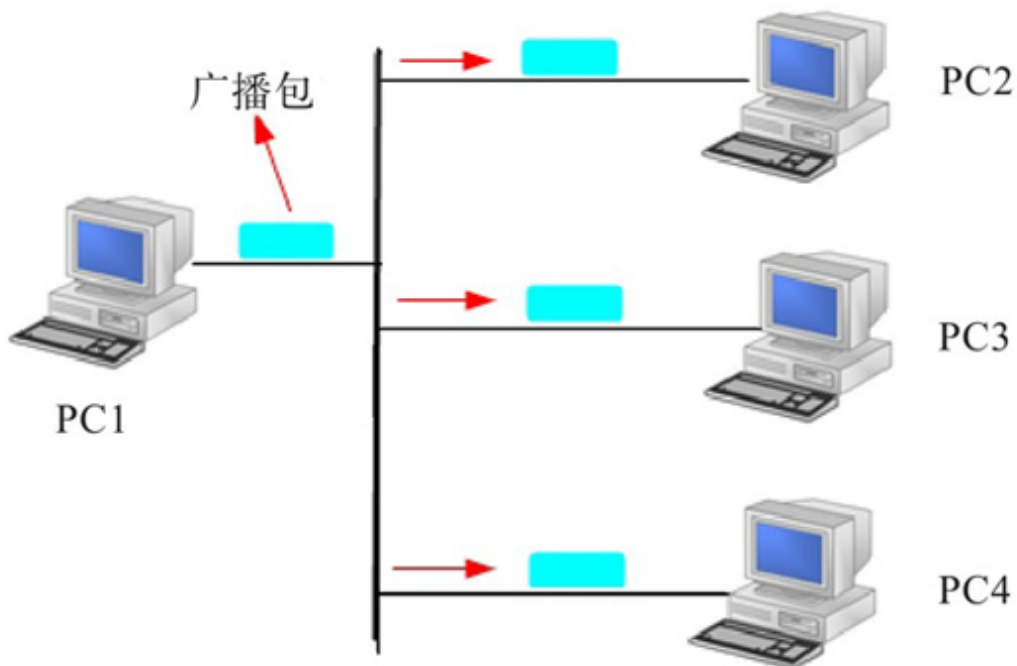


图 2-13 广播式传输网络示例

在广播式传输网络中传输信息时，任何一个节点都可以发送数据包，通过公共信道（如交换机的背板矩阵，以及设备间的连接通道）或总线传送到网络中的其他计算机上。然后，这些计算机根据数据包中的目的MAC地址进行判断，如果自己的MAC地址与目的MAC地址匹配则接收（与此同时，发送节点就可以知道与目的IP地址对应的MAC地址，下次接收到包括同样目的IP地址的包时就不用再广播了），否则便丢弃它。

区分是哪种传输方式网络，最关键的是看它里面所用的通信协议。以太网就是典型的广播式传输网络，其所使用的就是各种以太网（Ethernet）协议。本章后面将要介绍的环形拓扑结构的令牌环网络和

总线型拓扑结构的令牌总线网络也是广播式传输网络，因为在这两种拓扑结构网络中，任何一个站点发送数据，其他站点都可以接收到，总线两端的计算机是存在公共传输通道的，这条公共传输通道就是那条总线。当然，不仅环形、总线型拓扑结构的网络可以是广播式网络，其他的像星型、树型、网状拓扑结构的计算机网络都可以是广播式传输网络，因为这些网络中都存在公共信道。各种无线网络、卫星传播网络也都是广播式传输网络的，因为它们的传输信道都是公用的。具体的网络拓扑结构将在下节介绍。

2.3 计算机网络拓扑结构

拓扑（Topology）学是一种研究与大小、距离无关的几何图形特性的方法。“网络拓扑结构”是由网络节点设备和通信介质通过物理连接所构成的逻辑结构图。网络拓扑结构是从逻辑上表示网络服务器、工作站的网络配置和互相之间的连接方式和服务关系。在选择拓扑结构时，主要考虑的因素有：不同设备所担当的角色（或者设备间服务的关系）、各节点设备工作性能要求、安装的相对难易程度、重新配置的难易程度、维护的相对难易程度、通信介质发生故障时受到影响的设备的情况。

本节要分别介绍计算机网络（包括局域网和广域网）中的一些主要拓扑结构。在此先介绍与网络拓扑结构有关的几个基本概念。

2.3.1 网络拓扑结构相关基本概念

在设计网络拓扑结构时，我们经常会遇到如“节点”、“结点”、“链路”和“通路”这四个术语。它们到底各自代表什么，它们之间又有什么关系呢？

（1）节点

一个“节点”其实就是一个网络端口。节点又分为“转节点”和“访问节点”两类。“转节点”的作用是支持网络的连接，它通过通信线路转接和传递信息，如交换机、网关、路由器、防火墙设备的各个网络端口等；而“访问节点”是信息交换的源点和目标点，通常是用户计算机上的网卡接口。如我们在设计一个网络系统时，通常所说的共有××个节点，其实就是在网络中有多个要配置IP地址的网络端口。

(2) 结点

一个“结点”是指一台网络设备，因为它们通常连接了多个“节点”，所以称之为“结点”。在计算机网络中的结点又分为链路结点和路由结点，它们就分别对应的是网络中的交换机和路由器。从网络中的结点数多少就可以大概知道你的计算机网络规模和基本结构了。

(3) 链路

“链路”是两个节点间的线路。链路分物理链路和逻辑链路（或称数据链路）两种，前者是指实际存在的通信线路，由设备网络端口和传输介质连接实现；后者是指在逻辑上起作用的网络通路，由计算机网络体系结构中的数据链路层标准和协议来实现。如果链路层协议没有起作用，数据链路也就无法建立起来。

(4) 通路

“通路”从发出信息的节点到接收信息的节点之间的一串节点和链路的组合。也就是说，它是一系列穿越通信网络而建立起来的节点到节点的链路串连。它与“链路”的区别主要在于一条“通路”中可能包括多条“链路”。

2.3.2 星型拓扑结构

星型拓扑结构（Star Topology）又称集中式拓扑结构，是因集线器或交换机连接的各节点呈星状（也就是放射状）分布而得名。在这种拓扑结构的网络中有中央结点（集线器，或交换机），其他节点（工作站、服务器）都与中央结点直接相连。

1.基本星型拓扑结构单元

星型拓扑结构是目前应用最广、实用性最好的一种拓扑结构，这主要是因为它非常容易实现网络的扩展。无论在局域网中，还是在广域网中都可以见到它的身影，但其主要还是应用于有线以太网局域网中。所以事实上，星型拓扑结构主要应用于以太网局域网中，以太网包括许多标准，对应的标准集就是IEEE 802.3，具体将在第6章介绍。“星型拓扑结构”其实只是一个结构单元（一台集线器，或者交换机设备就是一个星型结构单元），多个星型结构单元连接起来又可以形成下面将要介绍的“树型拓扑结构”。图2-14所示的是最简单的单台集线器或交换机星型拓扑结构单元。

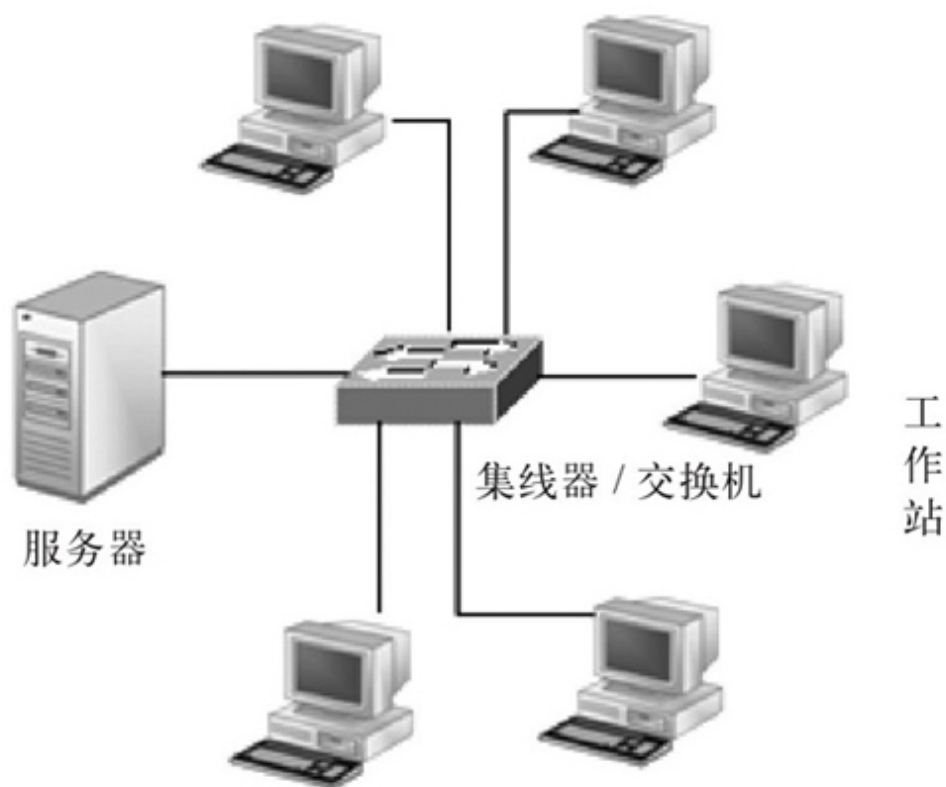


图 2-14 基本星型拓扑结构单元示例

在这个星型拓扑结构单元中，所有服务器和工作站等网络设备都集中连接在同一台交换机上。因为现在的固定端口交换机最多可以有48个或以上交换端口，所以这样一个简单的星型网络完全可以适用于用户节点数在40个以内的小型企业，或者分支办公室选用。模块式的交换机端口数可达100个以上，可以满足一个小型企业的需求。但实际上这种连接方式是比较少见的，因为单独用一台模块式的交换机连接成本要远高于采用多台低端口密度的固定端口交换机级联方式。模块式交换机通常用于大中型网络的核心层（或骨干层）或汇聚层，小型网络很少使用。

扩展交换端口的另一种有效方法就是堆叠了。有一些固定端口配置的交换机支持堆叠技术，通过专用的堆叠电缆连接，所有堆叠在一起的交换机都可作为单一交换机来管理，不仅可以使端口数量得到大幅提高（通常最多堆叠8台），还可以提高堆叠交换机中各端口实际可用的背板带宽，提高了交换机的整体交换性能。

有关交换机的级联和堆叠技术及配置的具体内容可参见笔者的《Cisco/H3C交换机配置与管理完全手册》（第2版）一书。

2.多级星型拓扑结构

复杂的星型结构网络就是在图2-14基础上通过多台交换机级联形成的，从而形成多级星型拓扑结构，满足更多、不同地理位置分布的用户连接和不同端口带宽需求。其实这就是下面将要介绍的“树型拓扑结构”。图2-15是一个包含两级交换机结构的星型网络，其中的两层交换机通常为不同档次的，可以满足不同需求。核心（或骨干层）交换机要选择档次较高的，用于连接下级交换机、服务器和有高性能需求的工作站用户等，下面各级则可以依次降低要求，以便最大限度地节省投资。

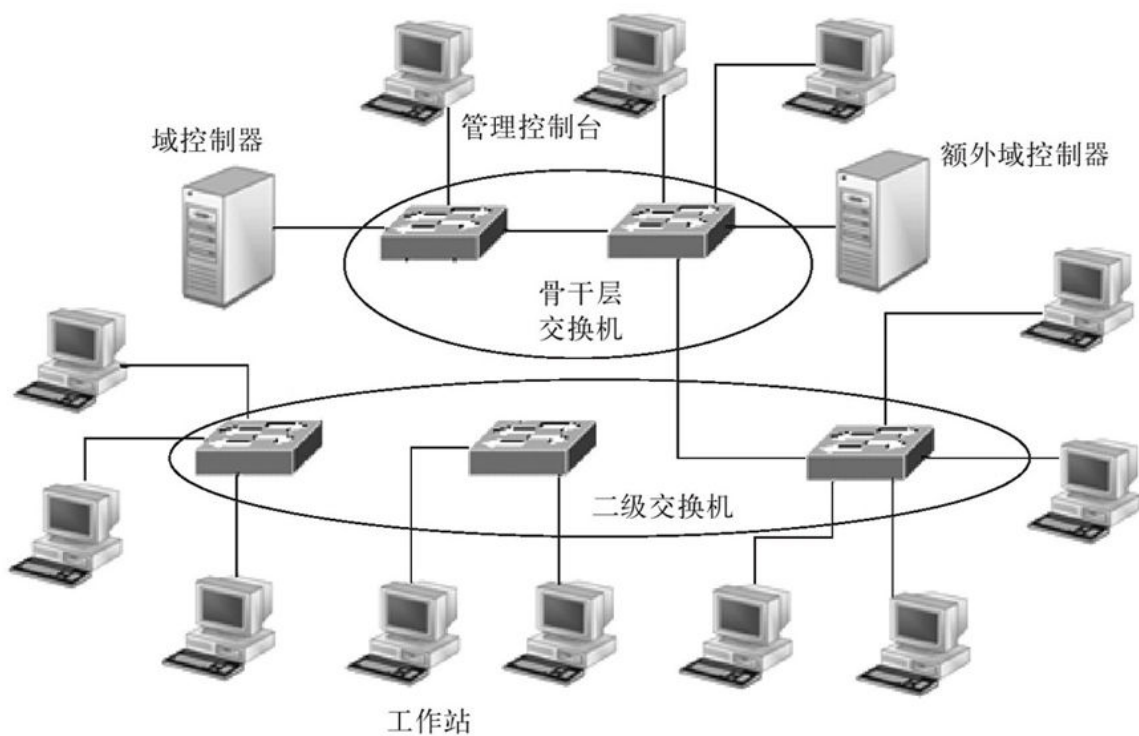


图 2-15 两级星型拓扑结构示例

当然，在实际的大中型企业网络中，其网络结构可能要比图2-15所示的网络复杂许多，还可能有三级，甚至四级交换机的级联（通常最多部署四级），还可能有交换机的堆叠和集群。图2-16所示网络结构中SS3 Switch 4400位置就是由两台这样的交换机堆叠组成的。

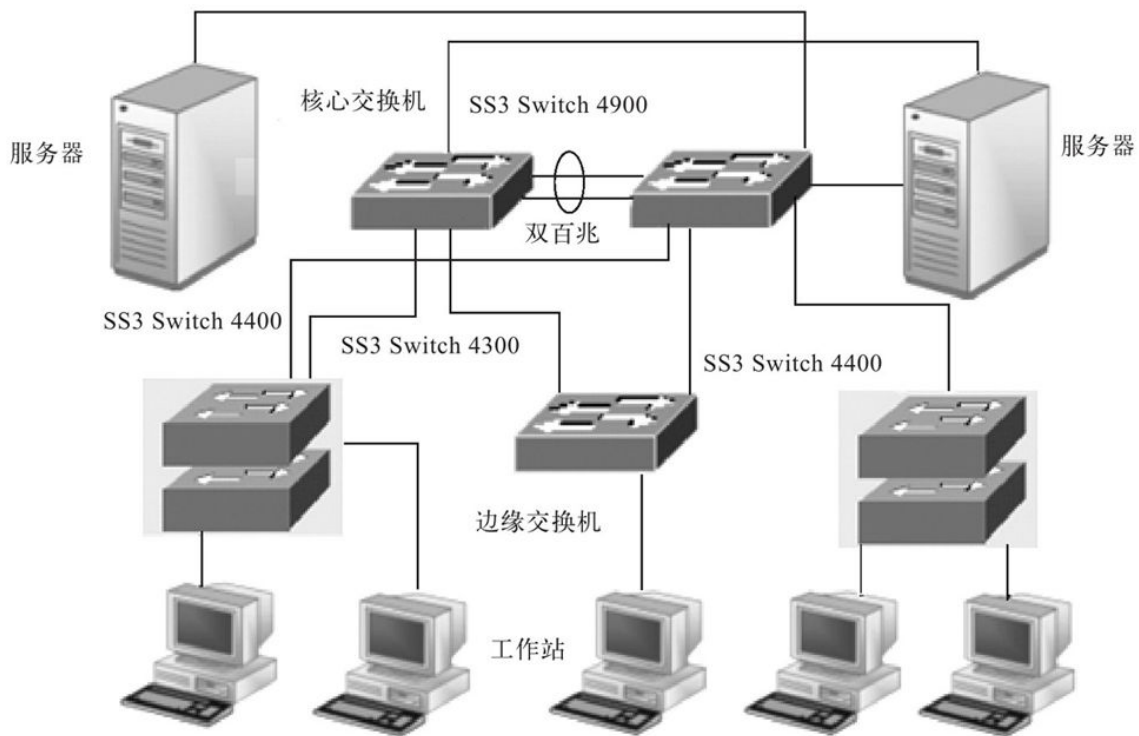


图 2-16 包含交换机堆叠的星型拓扑结构示例

3.星型拓扑结构传输距离限制

因为在星型结构网络中通常是采用双绞线和光纤作为传输介质的，而单段双绞线的最大长度为100m，集线设备放置在中心点，这样每一个采用此种结构的集线设备所能连接的网络范围最大直径就达到200m，超过这个范围都将要采用级联或中继的方法。采用光纤作为传输介质时虽然传输距离可以长许多，但也是有限制的。各种连接电缆长度限制如表2-1所示，1000BASE-SX网络的光纤长度限制如表2-2所示。

表 2-1 各种以太网电缆长度限制

以太网标准	电缆类型	单段最大长度 /km	网络接口类型
10BASE-T	3、4、5 类 100 欧 UTP（非屏蔽双绞线）	0.1	RJ-45
100BASE-T	5 类 100 欧 UTP	0.1	RJ-45
1000BASE-SX	50/125 或 62.5/125 微米多模光纤（MMF）	参见表 2-2	SC 或 ST
1000BASE-LH	9/125 微米单模光纤（SMF）	70	SC 或 ST
1000BASE-LX	9/125 微米 SMF	5	SC 或 ST
1000BASE-T	5 类、超 5 类或 6 类 UTP	0.1	RJ-45

表 2-2 1000BASE-SX 光纤长度限制

光纤直径 / μm	光纤带宽 / (MHz/km)	单段最大长度 /m
62.5/125 MMF	160	220
	200	275
50/125 MMF	400	500
	500	550

经验之谈 许多读者朋友认为，星型拓扑结构只适用于同楼层的网络，其实不是这样的。在多楼层，甚至多栋建筑物之间的网络互连多数也可以采用星型拓扑结构，因为它具有非常高的传输速率。从表2-1和表2-2可以看出，各楼层之间，各建筑物之间都可以采用普通的双绞线进行连接（通常是大对数的），只要距离在双绞线的有效距离范围内；当超过了双绞线的有效距离后，可以采用光纤连接（光纤的传输距离更远，传输性能更好），但光纤介质和相应接口的设备价格更贵，这必须予以充分考虑。同轴电缆只是一种低成本、低性能的选择，因为它的传输性能要远低于双绞线和光纤，在目前实际的楼层和建筑物之间网络互连很少采用。

4.星型拓扑结构主要优缺点

星型拓扑结构的优点主要体现在以下几个方面：

（1）节点扩展、移动方便

在星型拓扑结构网络中，节点扩展时只需要从交换机等集中设备空余端口中拉一条电缆与要加入的节点连接上即可；而要移动一个节点只需要把相应节点设备连接网线从设备端口拔出，然后移到新设备端口即可。上述过程并不影响其他任何已有设备的连接和使用，不会像下面将要介绍的环形网络那样“牵一发而动全身”。这是星型拓扑结构的最大优势。

（2）网络传输数据快

因为整个网络呈星型连接，网络的上行通道不是共享的，所以每个节点的数据传输对其他节点的数据传输影响非常小，这样就加快了网络数据传输速度。

另外，星型拓扑结构所对应的双绞线和光纤以太网标准的传输速率可以非常高（主要是因为相应的网络技术发展非常快），如普通的5类、超5类都可以通过4对芯线实现1000Mbps传输速率，7类屏蔽双绞线则可以实现10Gbps传输速率，光纤则更是可以轻松实现千兆、万兆的传输速率。而后面要介绍的环形、总线型结构中所对应的标准速率都在16Mbps以内，明显低了许多。

(3) 维护容易

在星型网络中，每个节点都是相对独立的，一个节点出现故障不会影响到其他节点的连接，可任意拆走故障节点。正因如此，这种网络结构受到用户的普遍欢迎，成为应用最广的一种拓扑结构类型。但如果集线设备出现了故障，也会导致整个网络的瘫痪。

星型拓扑结构的缺点主要体现在如下几个方面。

(1) 核心交换机工作负荷重

虽然说各工作站用户连接的是不同的交换机，但是最终还是要与连接在网络中央核心交换机上的服务器进行用户登录和网络服务器访问操作，所以中央核心交换机的工作负荷相当繁重，故对担任中央设备的交换机的性能和可靠性的要求非常高。其他各级集线器和交换机也连接多个用户，其工作负荷同样非常重，也要求具有较高的可靠性。

(2) 网络布线较复杂

每个计算机直接采用专门的电缆与集线设备相连，这样整个网络中至少就需要所有计算机及网络设备总量以上条数的电缆，这使得结构本就非常复杂的星型网络变得更加复杂了。特别是在大中型企业网络的机房中，太多的电缆无论对维护、管理，还是对机房安全都是一

个威胁。这就要求我们在布线时要多加注意，一定要在各条电缆、集线器和交换机端口上做好相应的标记。同时建议做好整个布线系统的标记和记录，以备日后出现布线故障时能迅速找到故障发生点。另外，由于这种星型网络中的每条电缆都是专用的，利用率不高，在较大的网络中，这种浪费还是相当大的。

(3) 广播传输影响网络性能

其实这是以太网的一个不足，但因星型拓扑结构主要应用于以太网中，所以相应的也就成了星型网络的一个缺点。因为在以太网中，当集线器收到节点发送的数据时，采取的是广播发送方式，任何一个节点发送信息在整个网中的节点都可以收到，这严重影响了网络性能的发挥。虽然说交换机具有MAC地址“学习”功能，但对于那些以前没有识别的节点发送来的数据，同样是采取广播方式发送的，所以同样存在广播风暴的负面影响。当然交换机的广播影响要远比集线器的要小得多，在局域网中使用影响不大。

综上所述，星型拓扑结构是一种应用广泛的有线局域网拓扑结构，特别是它可以采用廉价的双绞线进行布线，而且是非共享传输通道，传输性能好，节点数不受技术限制，扩展和维护容易，所以它又是一种经济、实用的网络拓扑结构。但受到单段双绞网线长度必须在100m以内的限制，超过这个距离则需要采取交换机级联拓展方式，或

者采用成本较高的光纤作为传输介质（不仅是传输介质的改变，相应设备也要有相应的接口）。

2.3.3 环形拓扑结构

环形拓扑结构（Ring Topology）在20世纪90年代计算机网络刚开始进入国内时采用得比较多，应用的标准是IEEE 802.5。可以说，令牌环在物理上是一个由一系列环接口（称之为中继转发器，即RPU）和这些接口间的点对点链路构成的闭合环路，各站点PC通过环接口连到网上。目前这一网络拓扑结构形式已不用了，因为它的传输速率最高只有16Mbps，扩展性能又不好，早已被性能远超过它的星型拓扑结构双绞线以太网替代了。

1. 环形网络结构概述

环形网络拓扑结构主要应用于采用同轴电缆作为传输介质的令牌网中，图2-17就是一个典型的环形网络。笔者曾于2000年为一家小型公司组建过局域网，该局域网采用的就是这种结构的。这种网络中的每一站点都是通过环中继转发器与它左右相邻的站点串行连接起来的，在传输介质环的两端各加上一个阻抗匹配器（又称终端匹配器）就形成了一个封闭的环路，“环形”结构的命名起因就在于此了。在细同轴电缆环形网中的环中继转发器是一个BNC接头，阻抗匹配器上的那个链子样的东西接在PC外壳上（相当于接地），如图2-18所示。

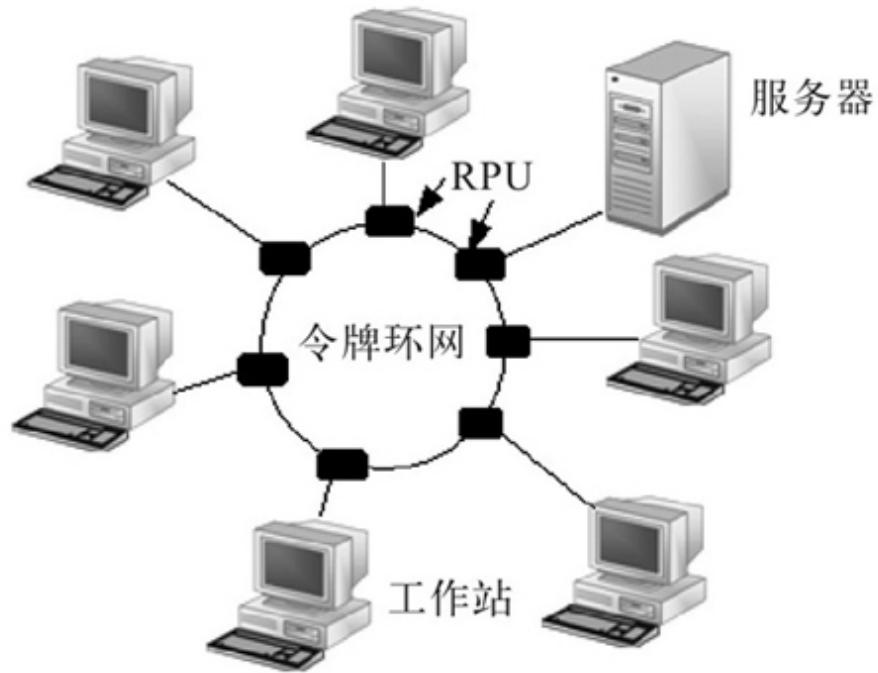


图 2-17 环形拓扑结构网络示例



图 2-18 BNC中继转发器和阻抗匹配器

说明 图2-17只是一种示意图，实际这种拓扑结构的网络不会是所有计算机真的要连接成物理上的环形，其连成的可以是任意形状，如

直线形、半环形等。这里所说的“环”是从电气性能上来讲的，“环”的形成并不是通过电缆两端直接连接形成的，而是通过在环的电缆两端加装一个阻抗匹配器来实现的。

环形拓扑结构网络的一个典型代表就是采用同轴电缆作为传输介质的IEEE 802.5的令牌环网（Token Ring Network）。令牌环拓扑结构最早是由IBM推出的，传输速率为4Mbps或16Mbps，比当时只有2Mbps的以太网性能要高出好几倍，所以在当时得到了广泛的应用。但随着以太网技术的跳跃式发展，令牌环网技术性能不能再适应时代的要求了，故逐渐被淘汰出局了。令牌环网的传输原理是，RPU（环中继转发器）从其中的一个环段（称为上行链路）上获取帧中的每个比特位信号，然后经过整形和放大转发到另一环段（称为下行链路）。如果帧中的目的MAC地址与本站点MAC地址一致，则复制该MAC帧发送给连接本RPU的站点。

2.令牌环网数据传输原理

令牌环网中由点对点链路构成的环路虽然不是真正意义上的广播媒体，但令牌环网上运行的数据帧仍能被所有的站点接收到。而且任何时刻仅允许一个站点发送数据，因此理论上是存在发送权竞争问题的，这实际上就是我们将在第6章介绍的“介质争用”问题。为了解决这个问题，在令牌环网中使用了一个称之为“令牌”（Token，可以联想到以前武将出征杀敌时手中拿的由皇上发的帅印，代表一种授权）的特

殊的MAC控制帧（该帧中有一个比特位用来标志令牌的忙或闲），使其沿着环路循环。并且规定只有获得“令牌”的站点才有权发送数据帧（就像战争中只有一位统帅可以获得“帅印”一样），完成数据发送后立即释放令牌以供其他站点使用。由于环路中只有一个令牌，因此任何时刻至多只有一个站点发送数据，不会产生冲突。但令牌环网上各站点均有相同的机会获得令牌。

图2-19所示的是一个简单令牌环网（注意，这里的环是逻辑意义上的，并不是真正的物理环），其中连接了4台机器。现假设A站点要向C站点发送数据。下面介绍A站点的具体数据发送流程。

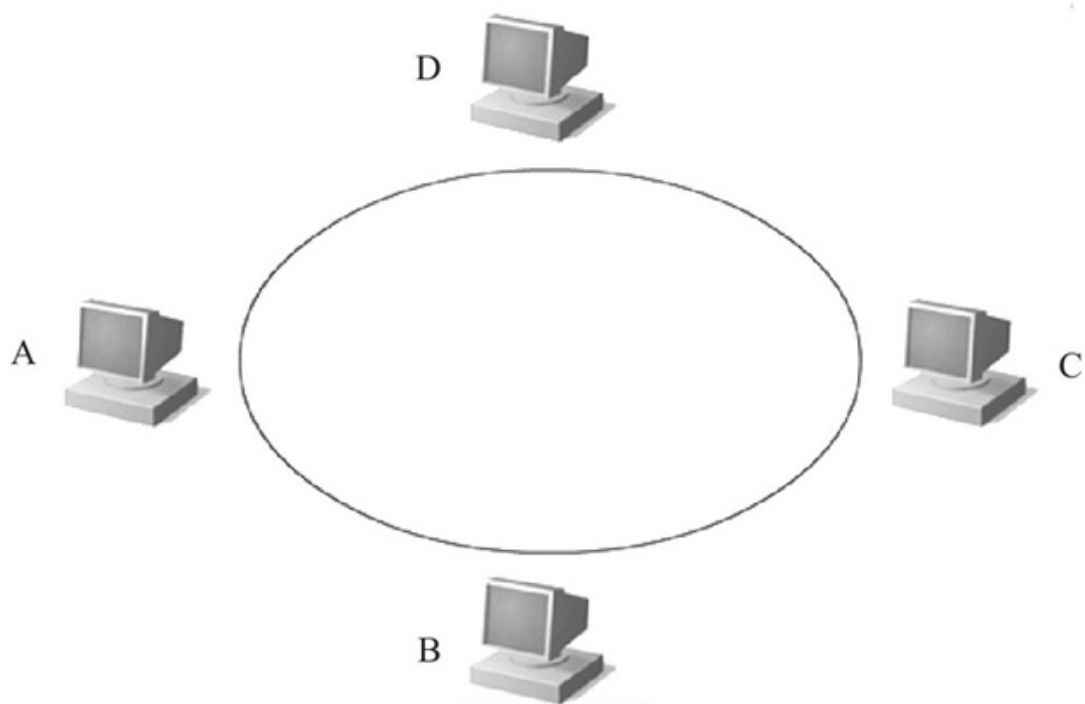


图 2-19 令牌环网示例

1) 首先环网中的令牌是在网上按照一个方向循环流动的，如图2-20所示（图中的“T”代表的就是环网中的“令牌”）。

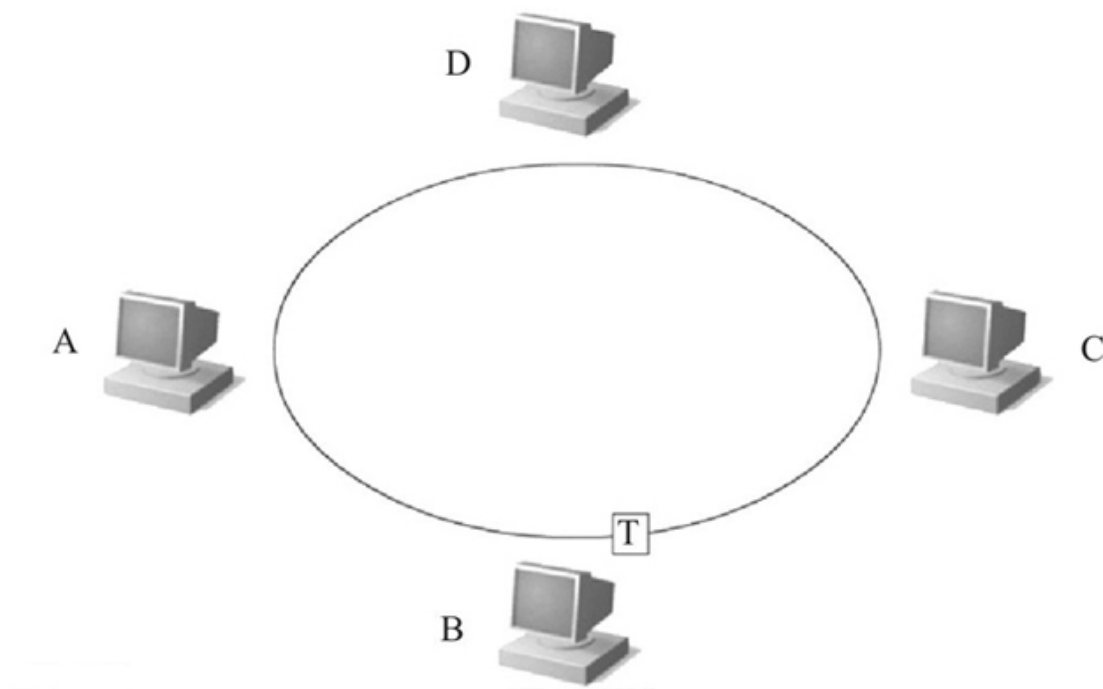


图 2-20 令牌在环网上流动

2) 当令牌转到A站点时，A站点的RPU截获该令牌（如图2-21所示），把令牌的状态控制位设置为1，代表处于忙的状态（表示令牌目前已被占用），同时在令牌帧上附加要发送的数据帧沿着环的路径向下游发送出去。当令牌帧和数据帧组合的信息帧（在此称之为“信息帧”）流经B站点时，B站点把自己的MAC地址与帧中的目的MAC地址进行比较，发现不匹配，于是不接收该数据帧，继续转发信息帧，如图2-22所示。

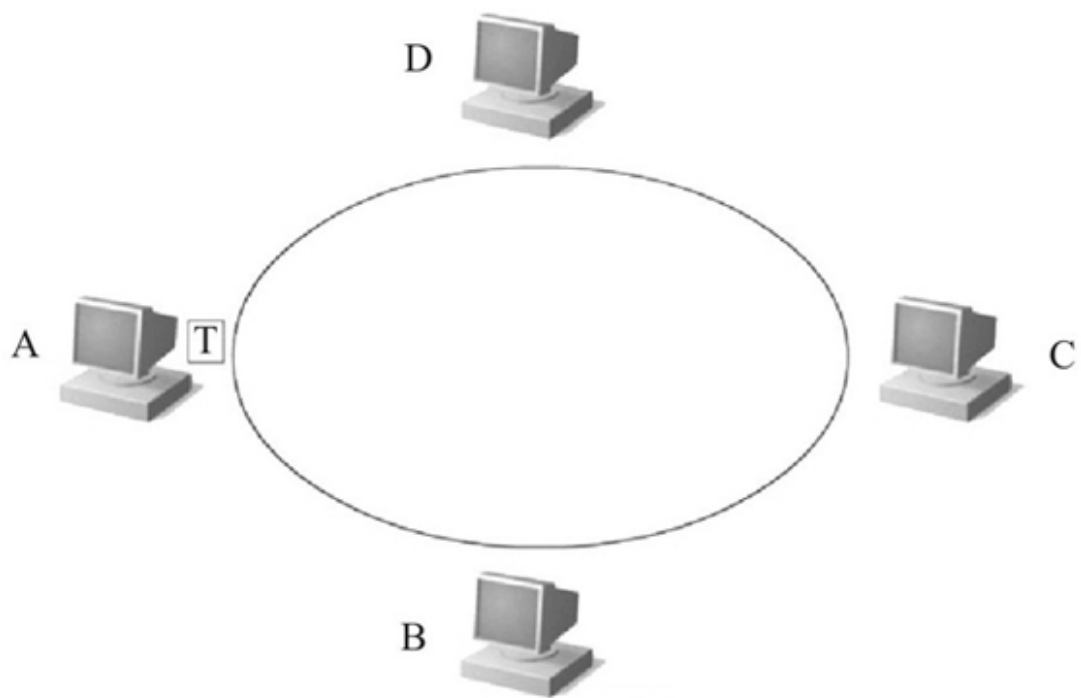


图 2-21 A站点截获令牌

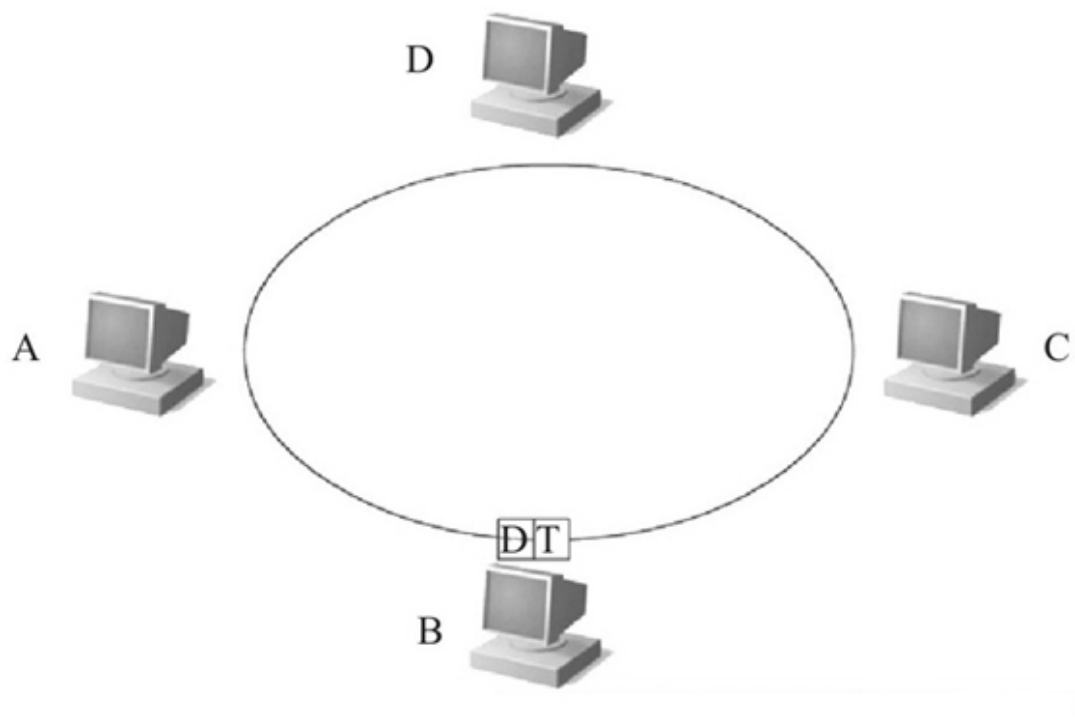


图 2-22 B站点不接收信息帧

3) 当信息帧流经C站点时, C站点也会把自己的MAC地址与帧中的目的MAC地址进行比较, 发现正好匹配, 于是C站点的RPU复制其中的数据帧, 并把复制的数据帧传送给C站点, 丢弃其中的令牌帧, 如图2-23所示。

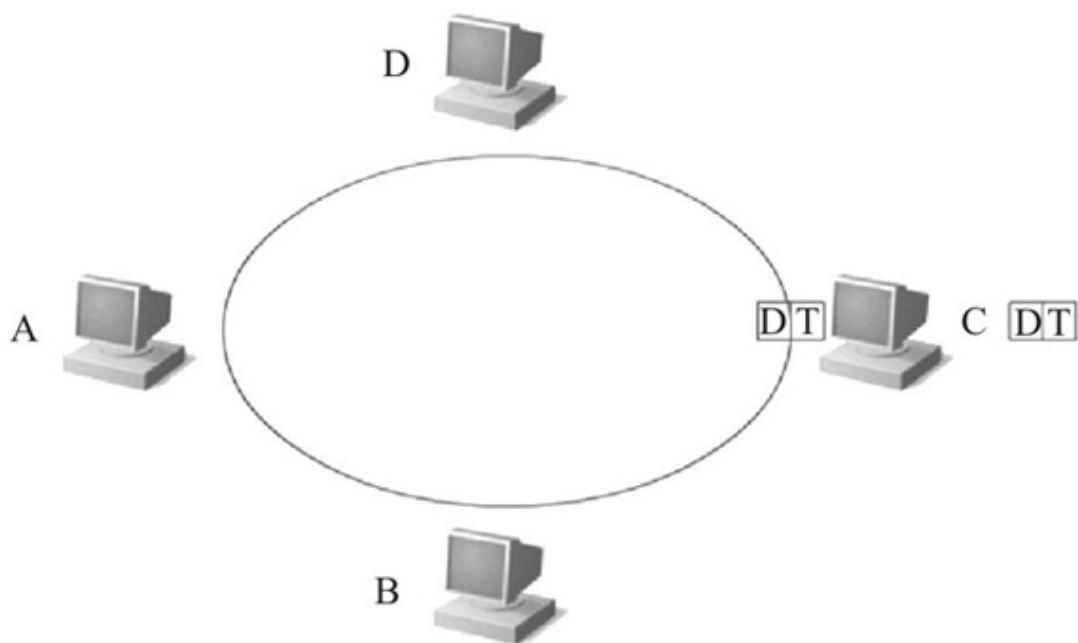


图 2-23 C站点的RPU复制该数据帧, 并把其传送给C站点

4) 原来的信息帧继续向下传递, 流经D站点时同样因它的MAC地址与数据帧中的目的MAC地址不匹配, 故不接收, 信息帧继续转, 直到回到源站点A, 如图2-24所示。

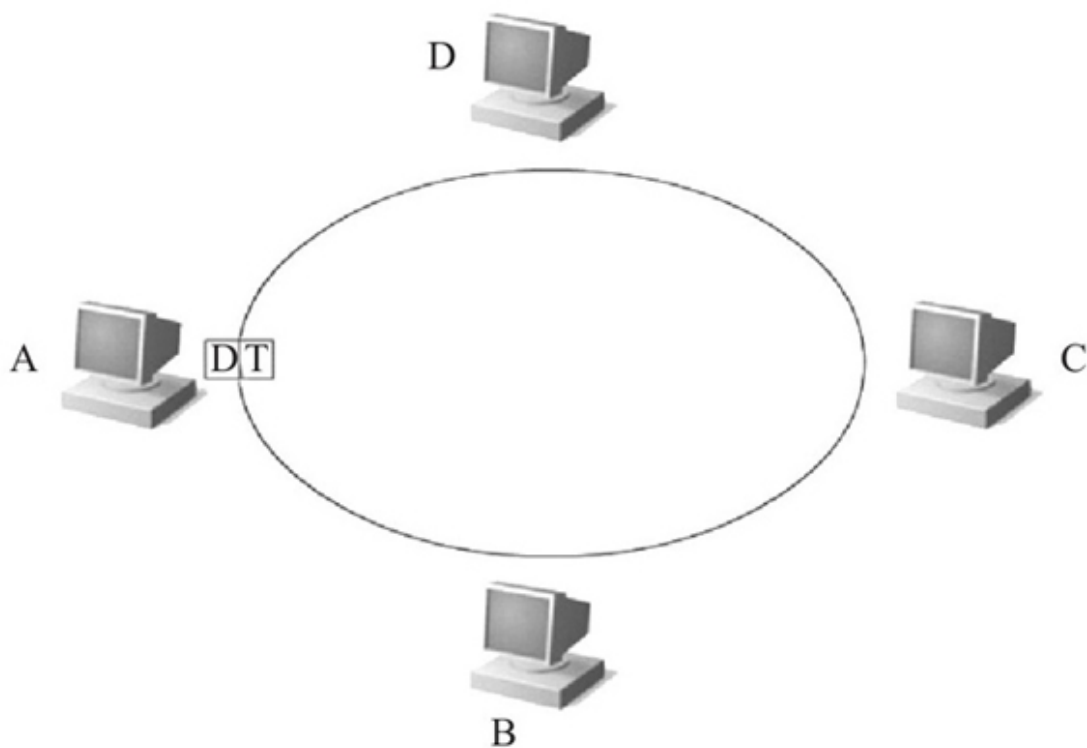


图 2-24 信息帧返回源站点，并释放令牌

此时恰好循环一周，A站点根据返回的有关信息确定所传数据帧有无出错。若有错则重发存于缓冲区中的待确认帧，否则释放缓冲区中的待确认帧。确认无错后，A站点的RPU会把其中的令牌帧状态置为0，表示处于空闲状态，释放令牌，然后在网上发送出去，令牌继续在环网上流动。其他站点如果想要发送数据，则可通过获得该令牌进行数据发送。

一个控制了令牌的站点可以进行一次或多次数据帧发送，只要不超过网络中规定的最长令牌控制时间即可。而具体的最长令牌控制时间要视具体网络环境通过计算才能得出，在此不做具体介绍。

3.环形拓扑结构的主要优缺点

环形拓扑结构网络的优点主要体现在以下方面。

(1) 网络路径选择和网络组建简单

在这种拓扑结构网络中，信息在环形网络中流动是沿着一个特定的方向，每两台计算机之间只有一个通路，简化了路径的选择，路径选择效率非常高。同样因为这个，这类网络的组建相当简单。

(2) 投资成本低

在投资成本方面，主要体现在令牌环网络中没有任何其他专用网络设备（如交换机），各站点直接通过电缆连接，所以无需任何投资去购买网络设备。

尽管有以上两个看似非常诱人的优点，但环形网络的缺点仍是主要的，这也是它最终被淘汰出局的根本原因。环形拓扑结构网络的缺点主要体现在以下几个方面。

(1) 传输速度慢

这是它最终不能得到发展和用户认可的最根本原因。虽说它在刚出现时，较当时的10Mbps以太网在速度上有一定优势（因为它可以实现16Mbps的接入速率），但由于这种网络技术后来一直没有任何发

展，速度仍保留在原来水平，相对现在最起码的100Mbps、1Gbps速率的以太网来说，实在是太落后了。现在连无线局域网（WLAN）的传输速度都远远超过了它。这么低的连接性能决定了它只能被淘汰的局面。

（2）连接用户数非常少

在这种环形拓扑结构中，各用户是相互串联在一条传输电缆上的，所以可以连接的用户数非常有限。尽管可以有中继设备，但中继器只起到一个信号放大和连接距离的拓展的作用，并不能很明显地提高连接用户的数量（通常最多也就是几十个用户）。

（3）传输效率低

这种环形拓扑结构网络共享一条传输电缆，每发送一个数据均要先取得令牌，每次数据的发送，令牌都要在整个环状网络中从头走到尾，哪怕是已有站点接受了数据；而且每个环形网络中只有一个令牌，所以同一时刻只有一个站点可以取得令牌并发送数据，所以传输效率是非常低的，明显不再适用当前复杂的网络应用需求。

（4）扩展性能差

因为是环形拓扑结构，且没有任何可用来扩展连接的设备，决定了它的扩展性能远不如星型拓扑结构好。如果要新添加或移动站点，

就必须中断整个网络，在适当位置切断网线，并在两端做好环中继转发器才能连接。

（5）维护困难

虽然在环形拓扑结构网络中只有一条传输电缆，看似结构也非常简单，但它是一个闭环，设备都连接在同一条串行连接的环路上，所以一旦某个站点出现了故障，整个网络将瘫痪。并且在这样一个串行结构中，要找到具体的故障点非常困难，必须一个站点一个站点地排除，非常不便。另一方面，因为同轴电缆所采用的是插针接触方式，也很容易出现接触不良，造成网络中断，网络故障率非常高。笔者就曾经维护过这样一个小型企业网，虽然只有20多台机，但因分布在几栋建筑物中，几乎天天发生网络故障，有时一查就可能是几个小时。

2.3.4 总线型拓扑结构

总线型拓扑结构（Bus Topology）与上节介绍的环形拓扑结构从外形上看有些类似，都是共享一条同轴电缆作为传输介质，通过RPU（中继转发器）连接多台计算机，而且网络通信中都是以令牌的方式进行的。所谓“总线”就表示，网络中连接的各站点间进行数据通信时都必须通过这条线缆。但总线型拓扑结构采用的是IEEE 802.4标准（对应RFC 1230），接入速率也低于上节介绍的环形网络（只有10Mbps），这两种拓扑结构还是存在较大不同的，具体将在本节后面介绍。但要说明的是，在当前的局域网中，纯粹的总线型结构网络基本上不见了，取而代之的是在一些特殊的网络环境中与星型拓扑结构混合使用，也就是在本章后面即将介绍的混合型拓扑结构。

1.总线型结构概述

总线型拓扑结构网络中所有设备通过连接器并行连接到一条传输电缆（通常称之为中继线、总线、母线或干线）上，并在两端加装一个称为“终接器”的组件，如图2-25所示。终接器主要用来与总线进行阻抗匹配，最大限度吸收传送端部的能量，避免信号反射回总线产生不必要的干扰。

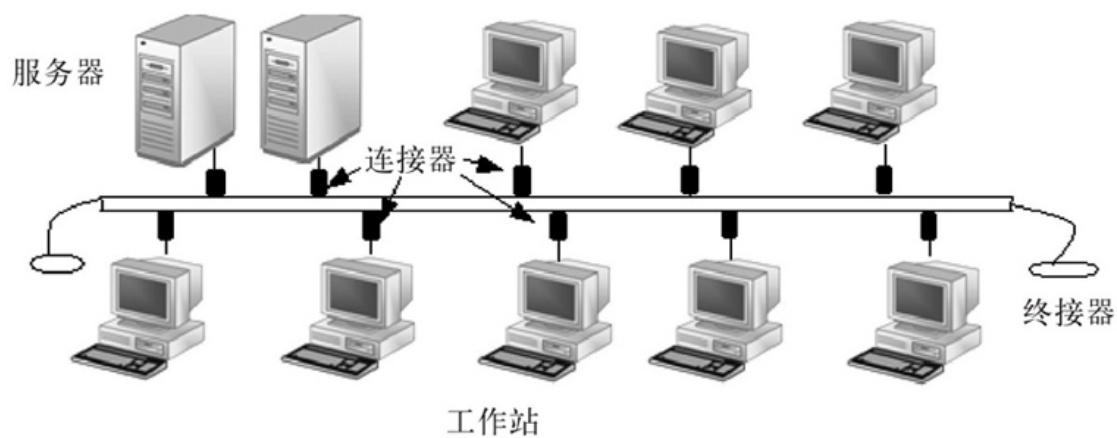


图 2-25 总线型结构示例

总线型结构网络所采用的传输介质一般也是同轴电缆（包括粗同轴电缆和细同轴电缆，也有采用光纤的），如ATM网、Cable Modem所采用的网络等都属于总线型网络结构。为了扩展所连接的计算机数量，可以在网络中添加其他的扩展设备，如中继器。图2-26所示的就是通过中继器连接的两个总线型网络单元。

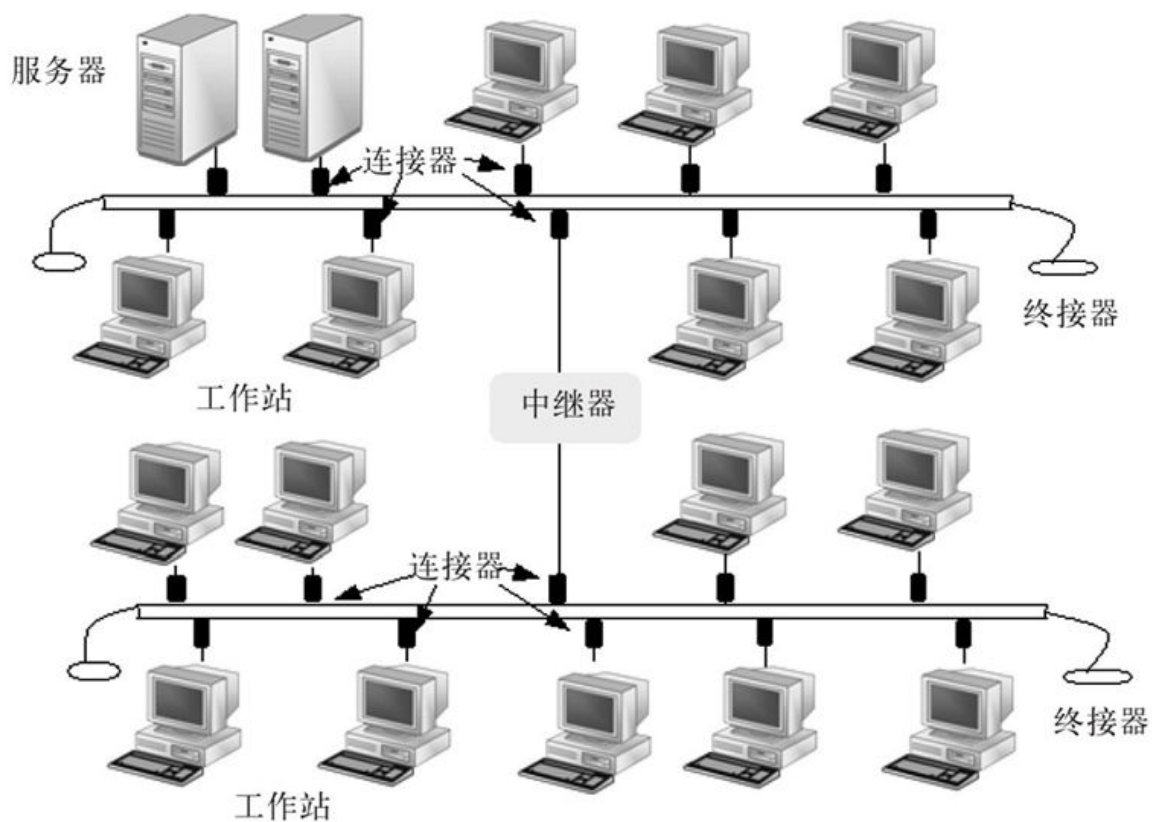


图 2-26 双总线结构网络互连示例

总线型拓扑结构的代表技术就是IBM的ARCNet令牌网络，所以总线型拓扑结构通常认为是令牌总线（Token Bus）结构。物理上是总线网，逻辑上是令牌网。总线型拓扑结构与上节介绍的环形拓扑结构相比，不同之处主要体现在以下两个方面。

（1）与传输电缆的连接方式不同

环形拓扑结构中的连接器与电缆是串联的，所以任何连接站点出现故障都会引起整个网络的中断，而总线型拓扑结构中的连接器与电

缆是并联的，所以某个站点发生故障不会影响网络中的其他站点通信。

（2）数据传输原理不同

两种拓扑结构的数据传输原理不一样。在总线型拓扑结构中，令牌帧和数据帧都是沿着根据当前网络环境自动生成的逻辑令牌环进行传输的，而不是像环形拓扑结构那样按照物理环路径进行传输的。具体将在本节后面介绍。

2.令牌总线数据传输原理

IEEE 802.4标准下的令牌总线的介质访问控制（MAC）是根据局域网物理总线的各站点先生成一个逻辑环，每一个站点都在一个序列中被指定一个逻辑位置（注意，不是物理位置），序列中最后一个站点的后面又跟着第一个站点，以形成一个逻辑上闭合的环路。图2-27所示的就是一个小型总线拓扑型拓扑结构计算机网络示例。从图中可以看出，在物理结构上，它是一个总线拓扑结构局域网，但是在逻辑结构上，又成了一种环形拓扑结构的局域网：

$A \rightarrow D \rightarrow C \rightarrow G \rightarrow F \rightarrow B \rightarrow E \rightarrow A$ 。

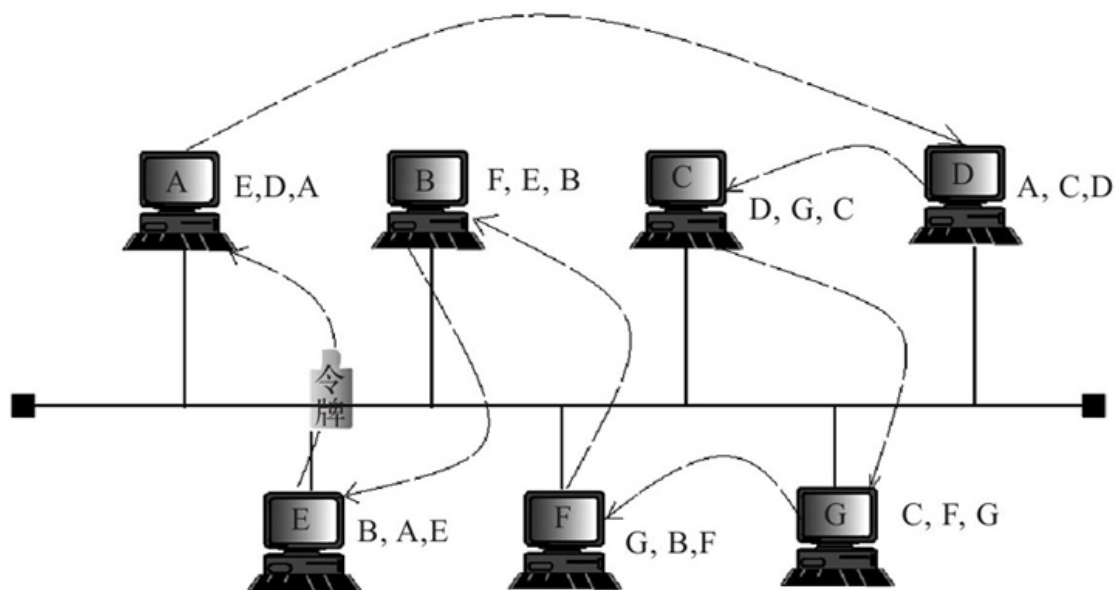


图 2-27 令牌总线中的站点连接表

另外，在总线型拓扑结构网络中，每个站点都知道在它之前的站点和在它之后的邻居站点标识。为了保证逻辑闭合环路的形成，每个站点都动态地维护着一个连接表，该表记录着本站点在环路中的前继、后继和本站点的地址。如图2-27所示，A站点中有E、D、A这三个地址，分别代表着A站点的前继站点（E站点）MAC地址、后继站点（D站点）MAC地址和本站点MAC地址。每个站点根据它的后继站点MAC地址确定下一个可能要占有令牌的站点。所以，令牌的传递顺序与总线上各站点的物理位置无关。

如果总线网络中某个站点出现故障或没有工作，则会重新建立新的逻辑令牌环网络，绕开出现故障或没有工作的站点（因为各站点之

间是并行连接的），如图2-28所示的就是图2-27所示的总线型拓扑结构网络在出现两个站点（C和G站点）没有正常工作时新建的令牌环网。

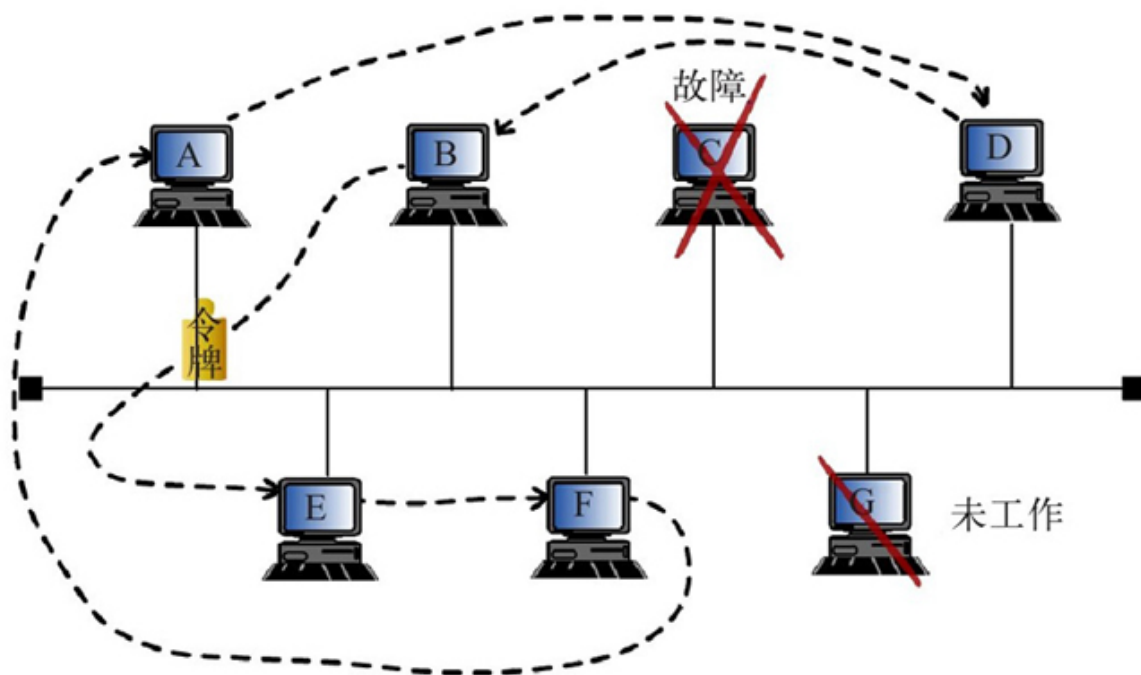


图 2-28 当总线网络中出现不能正常工作站点时新建的令牌环网络

令牌总线网络与令牌环网络的数据传输原理基本一样，都是站点在发送数据前必须先取得“令牌”，取得令牌的站点有数据帧要发送则可发送，如果没有数据帧要发送，则直接把令牌传递到逻辑令牌环中的后继站点。而且只有获得令牌的站点才能发送信息，其他站点只能接收信息，或者被动地发送信息（在拥有令牌的站点要求下发送信息）。由于站点接收到令牌的过程是顺序依次进行的，因此对所有站点都有公平的访问权，也不会出现介质访问冲突。

在具体的数据发送过程中，取得了令牌的站点把令牌帧和数据帧一起（在此也称之为信息帧）在总线网络中发送，其他各站点均可收到这个信息帧，但也只有其MAC地址与接收到的数据帧中的目的MAC地址一致的站点才会复制并接收该信息帧，把其中的数据帧传递给对应的站点PC（将其中的令牌帧丢弃），然后原来的信息帧继续沿着逻辑令牌环传递，直到回到发送数据的源站点。当发送数据帧的站点收到接收数据帧的站点的响应后就可得知该次数据发送成功，随即释放自己所控制的令牌，按照逻辑令牌环的顺序把令牌依次传递到它的后继站点。

3.总线拓扑结构的主要优缺点

总线拓扑结构的优点与环形拓扑结构差不多，主要有如下几点。

（1）网络结构简单，易于布线

因为总线型网络与环形网络一样都是共享传输介质，也通常无需另外的网络设备，所以整个网络结构比较简单，布线比较容易。

（2）扩展较容易

这是总线型网络相对同样是采用同轴电缆（或光纤）作为传输介质的环形网络结构的最大的一个优点。因为总线型结构网络中，各站点与总线的连接是通过并行连接（环形网络中连接器与电缆的连接是

串行的)实现的,所以站点的扩展无需断开网络,扩展容易了许多。而且还可通过中继设备扩展连接到其他网络中,进一步提高了可扩展性能。

(3) 维护容易

总线型网络中的连接器与总线电缆是并行连接的,这给整个网络的维护带来了极大的便利,因为一个站点的故障不会影响其他站点,更不影响整个网络,所以故障点的查找就容易了许多。这与星型拓扑结构类似。

尽管有以上一些优点,但是它与环形拓扑结构网络一样,缺点仍是主要的,这些缺点也决定了它在当前网络应用中也极少使用的命运。总线型结构的主要缺点表现在以下几个方面。

(1) 传输速率低

IEEE 802.5令牌环网中的最高传输速率可达16Mbps,但IEEE 802.4标准下的令牌总线网络标准最高传输速率仅为10Mbps。所以它虽然在扩展性方面较令牌环网络有一些优势,但它同样摆脱不了被淘汰的命运。

(2) 故障诊断困难

虽然总线拓扑结构简单、可靠性高，而且是互不影响的并行连接，但故障的检测仍然很不容易。这是因为这种网络不是集中式连接，故障诊断需要在网络中各站点计算机上分别进行。另外，在这种结构中，如果故障发生在各个计算机内部，只需要将计算机从总线上去掉，比较容易实现。但是如果是总线传输介质发生故障，则需要更换全部相应段传输介质了。

(3) 难以实现大规模扩展

虽然相对环形网络来说，总线型网络在扩展性方面有了一定的改善，可以在不断开网络的情况下添加设备，还可添加中继器之类的设备予以扩展，但受到传输性能的限制，其扩展性仍然远不如星型网络，难以实现大规模的扩展。

综上所述，单纯总线型结构网络目前也已基本不用，因为传输性能太低（只有10Mbps），可扩展性也受到性能的限制。目前总线型结构就是在后面将要介绍的混合型网络中才会用到的。在这些混合型网络中使用总线型结构的目的是用来连接两个（如两栋建筑物），或多个（如多楼层）相距超过100m的局域网，细同轴电缆连接的距离可达185m，粗同轴电缆可达500m。如果超过这两个标准，就需要用到光纤了。但无论采用哪种传输介质的总线型结构，传输速率都只有10Mbps，实用性极低，还不如直接采用光纤星型拓扑结构。

2.3.5 树形拓扑结构

关于树形拓扑结构（Tree Topology）的描述，目前有多种版本，有的说是总线型拓扑结构的扩展，有的说是星型拓扑结构的扩展，其实两者均有道理。之所以认为是星型拓扑结构的扩展，是因为其中的每个集线设备（如交换机）所连接的就是一个个星型拓扑结构单元。之所以认为是总线型拓扑结构的扩展，是因为树形拓扑结构中各设备间是通过类似的“总线”（交换机级联电脑）进行互连的，一个星型结构单元的节点与另一个星型结构单元的节点之间的通信都是共享这一条“总线”的。一般认为树形拓扑结构是星型拓扑结构的扩展。

树形拓扑结构自上而下（从核心交换机（或骨干层）到汇聚层，再到边缘层）是自上而下依次分层扩展的，就像一棵倒放的“树”，这或许就是把它定义为“树形”拓扑结构的原因之一吧。树形拓扑结构的最顶层（也就是核心层）相当于树的“根”，中间层（汇聚层）相对于“树的枝”，而最下层（边缘层，或者接入层）则相当于树枝上的“细枝”和“树叶”。自上而下，所用的交换机数量是逐层增多的。简化的树形拓扑结构如图2-29所示（图中每一个大圆圈代表一台交换机，最下面的每个小圆圈代表一台所连接的主机）。

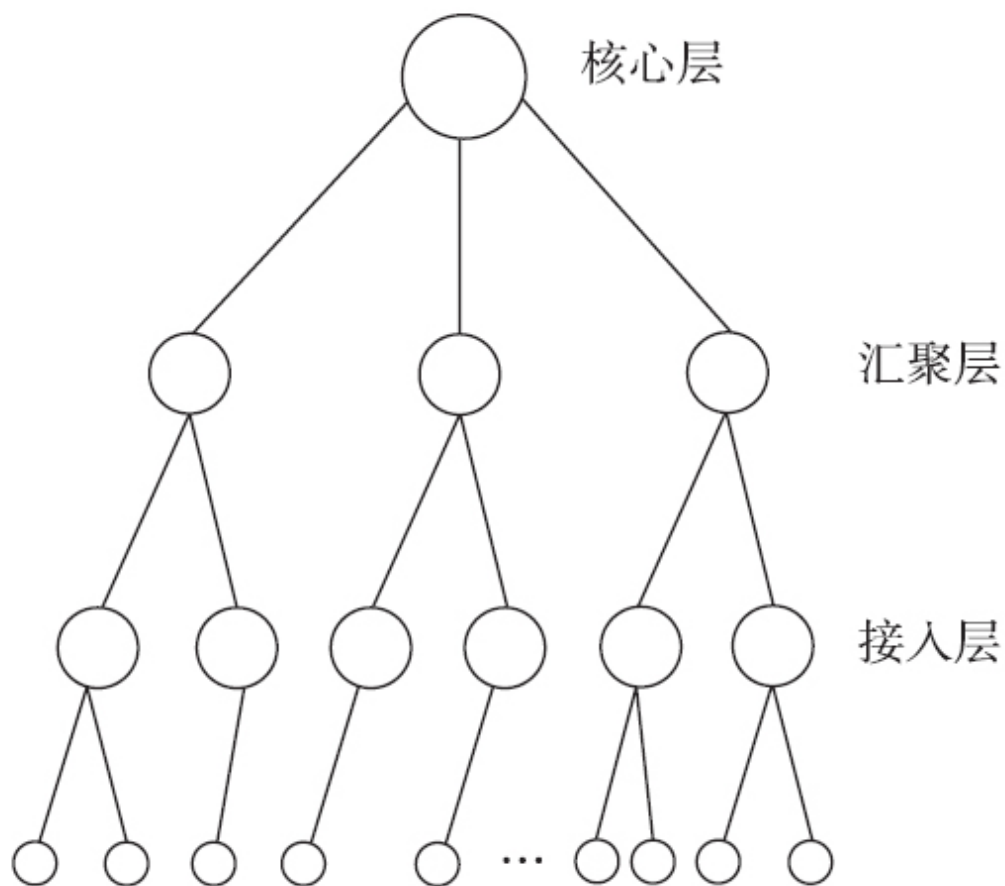


图 2-29 树形拓扑结构示意图

经验之谈 其实还有许多类似“树”形状的实例，如各公司的组织结构也是自上而下依次展开的（最上面只有董事长一人，下面是各级董事，再下面是总经理、部门经理、科室主任等），还有我们所用的Internet网站域名也是自上而下展开的（分为根域名、顶级域名（一级域名）、二级域名、三级域名等，如图2-30所示），这部分将在第11章具体介绍。

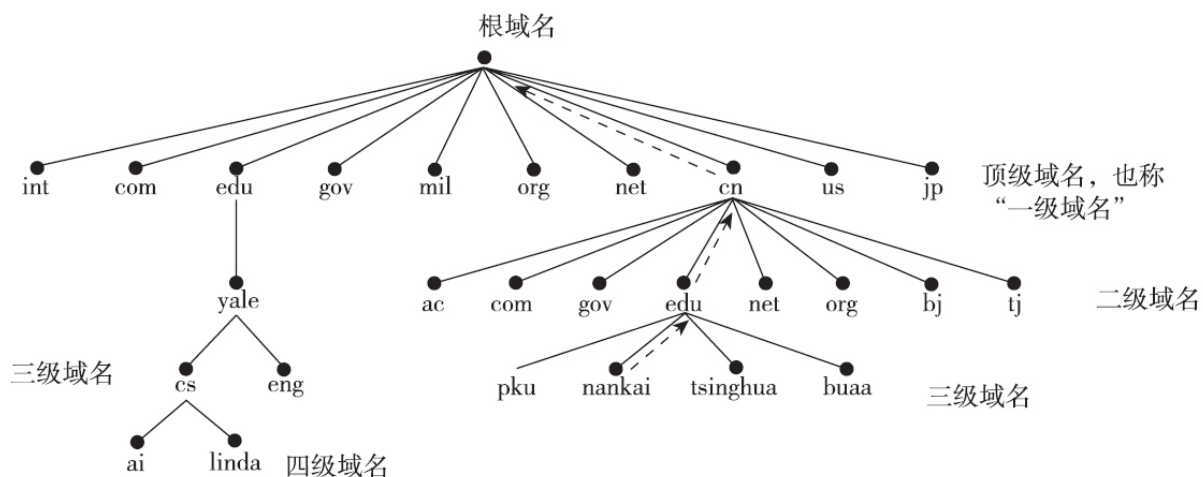


图 2-30 域名结构

在计算机网络技术中，还有一种技术同样引用了“树”的概念，那就是将在第6章介绍的生成树协议（STP）。它形象地利用了“树”中树根、树干、树枝、树叶之间这种无交叉的逻辑关系，以实现交换网络中无二层环路。

图2-31所示的就是一个典型的树形拓扑结构。它采用分级的集中控制方式，其传输介质可有多条分支，但不形成闭合回路，每条通信线路都必须支持双向传输。

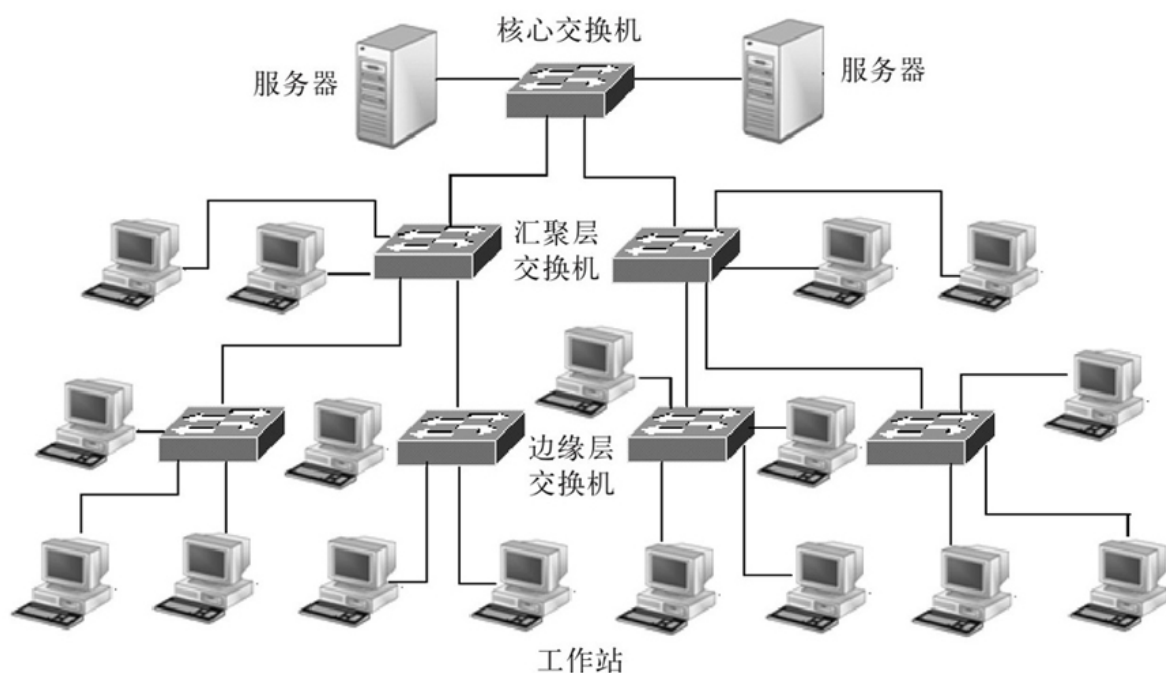


图 2-31 典型树形拓扑结构示例

从以上介绍可以得知，树形拓扑结构的主要优点还是扩展性方面。星型拓扑结构便于扩展我们已经知道，只要在集线设备空余端口上拉出一条网线，就可以添加新的设备；而在树形拓扑结构中，是通过多级星型结构的级联，可以更方便地实现在连接距离和端口数据上的扩展，实现更大规模网络的扩展升级。

但树形拓扑结构自身也有一些不足，这主要体现在以下两个方面：一是对“根”设备（核心，或者骨干层）交换机的依赖性太大，如果“根”发生故障，则那些依赖“根”设备访问的服务器或外网则全部不可访问了，相当于总线型拓扑结构中总线中断后，所有用户网络都中断一样。另外，处于最顶端的核心层设备，因为下面连接了更多的级联

设备和用户，负荷更重，需要配备性能更强的交换机和路由设备，成本比较高。但这些不足都可以通过配置冗余链路和选择高性能设备来弥补。树形拓扑结果是目目前中小型以太局域网（如位于同一楼层，或者分布于少数几个楼层的局域网）中最主要的拓扑结构。

2.3.6 网状拓扑结构

网状拓扑结构（**Mesh Topology**）又称无规则型拓扑结构。在这种结构中，各节点之间通过传输介质彼此互连，构成一个网状结构。

网状拓扑结构又有“全网状结构”和“半网状结构”两种。所谓“全网状结构”就是指网络中任何两个节点间都是相互连接的。假设一个网络中有 n 个节点，则任何一个节点就有 $n-1$ 条与其他节点的连接。图2-32所示的就是一个全网状拓扑结构。而所谓的“半网状结构”是指网络中并不是每个节点都与网络的其他所有节点有连接，可能只是一部分节点间有互连，如图2-33所示，A节点就没有与E节点直接连接，C节点也没有与F节点直接连接。

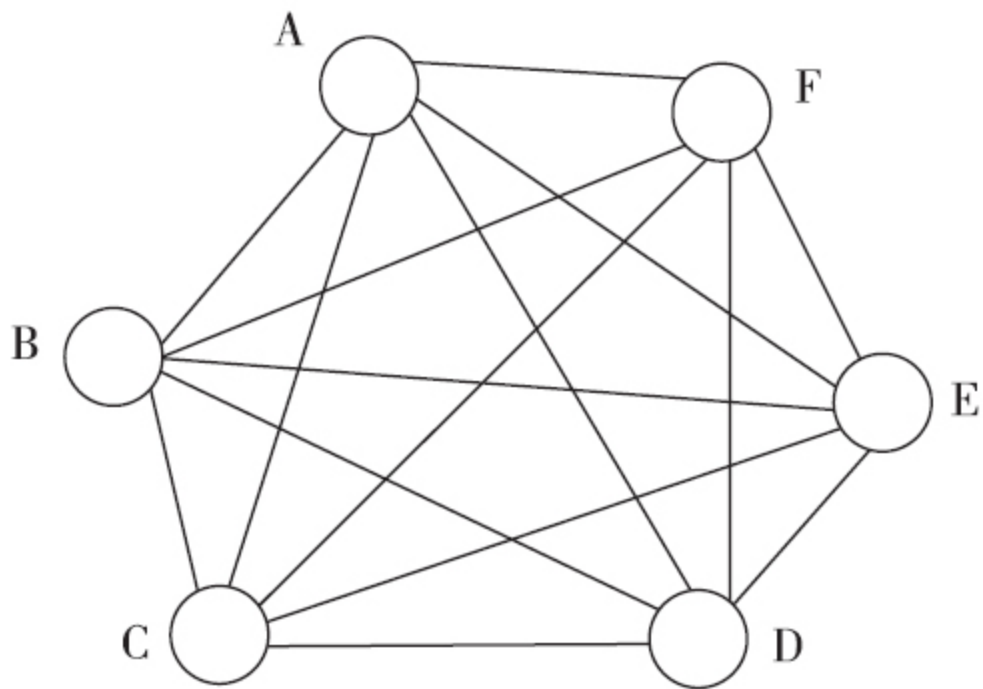


图 2-32 全网状拓扑结构示例

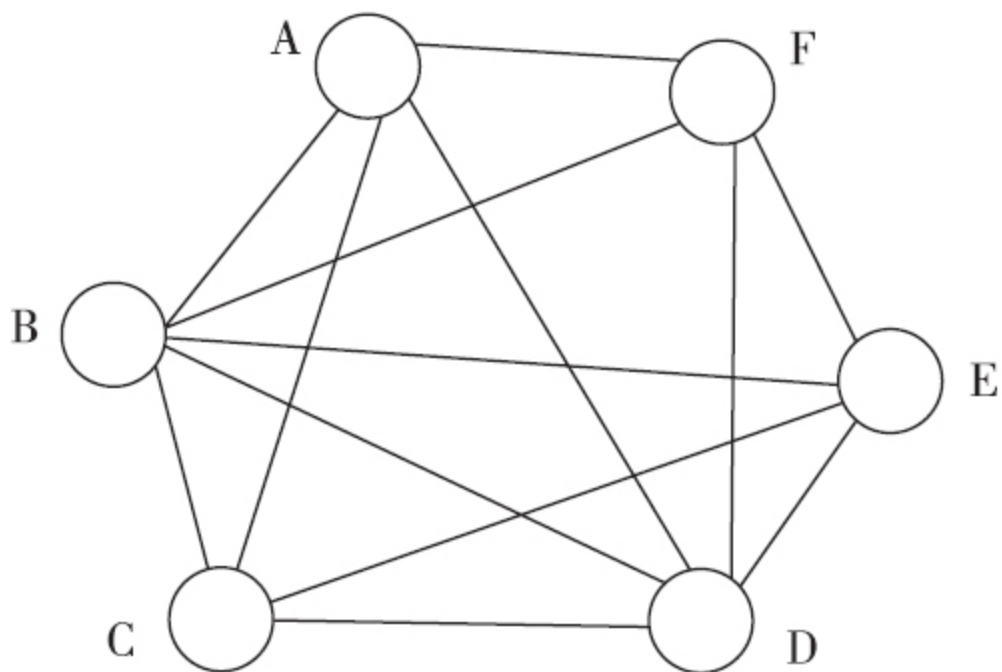


图 2-33 半网关拓扑结构示例

从以上介绍可以知道，网状拓扑结构的布线是相当复杂的（中间没有集中连接设备，全靠电缆来互连），布线成本也非常高，因为每个节点要用多条电缆与其他节点依次连接。同样，由于中间没有集中连接设备，每个节点PC都需要安装多块网卡，当一个节点要互连的其他节点比较多时，这显然不可行。所以这种网状拓扑结构在局域网中是极少使用的，最多也只是极少数的节点采用了半网状拓扑结构。

网状拓扑结构主要用于广域网中，这时它们连接的不再是终端用户PC节点，而是网络设备结点，如网络中的交换机、路由器等设备。在广域网中采用网状拓扑结构可以实现多个网段，或者子网间的彼此互连。因为交换机和路由器这些设备本身就具有多个网络端口，所以进行网状连接也很简单，只是需要多拉几条线而已。

广域网中采用网状拓扑结构的主要目的就是实现链路或路由线路的冗余，提高网络的可靠性。当然，一般并不会在整个广域网中而只是在骨干网络中采用这种拓扑结构。

图2-34所示为一个广域骨干网全网状拓扑结构示例。示例中各个路由器之间彼此互连。但更多的是采用半网状拓扑结构，仅少数结点需要与网络中其他所有结点互连。如图2-35所示的就是广域骨干网中采用半网状拓扑结构的一个示例。示例中各路由器只与少数其他路由器互连，并不是全部互连。

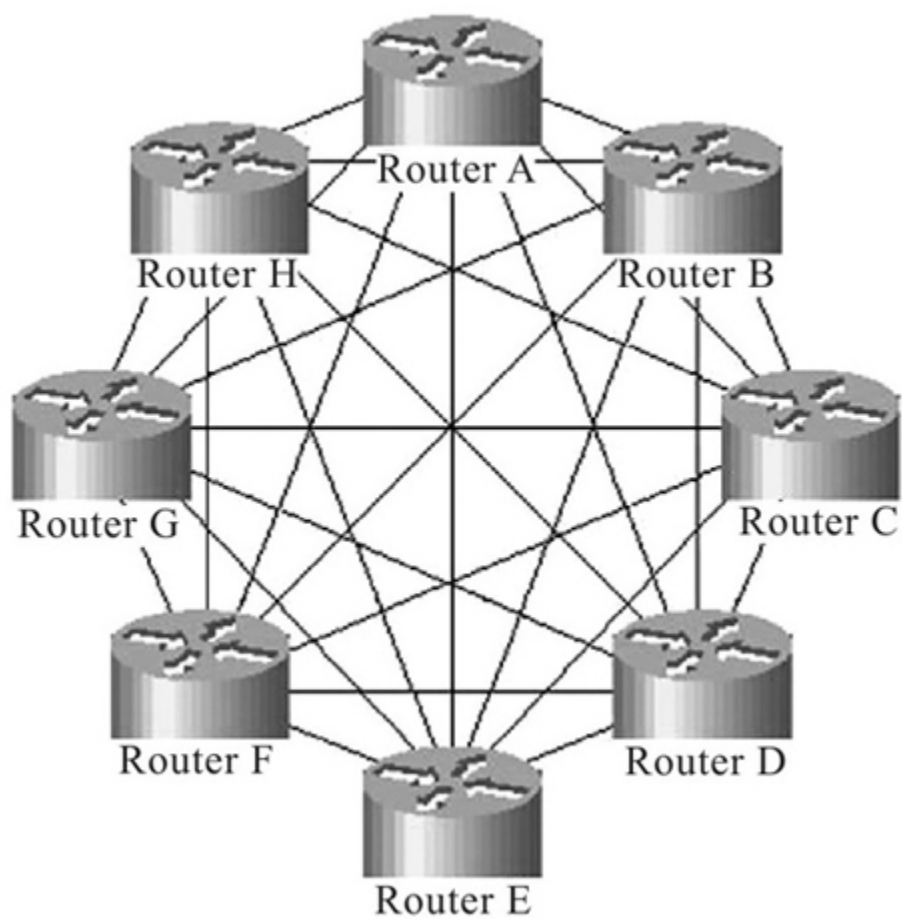


图 2-34 广域骨干网中的全网状拓扑结构示例

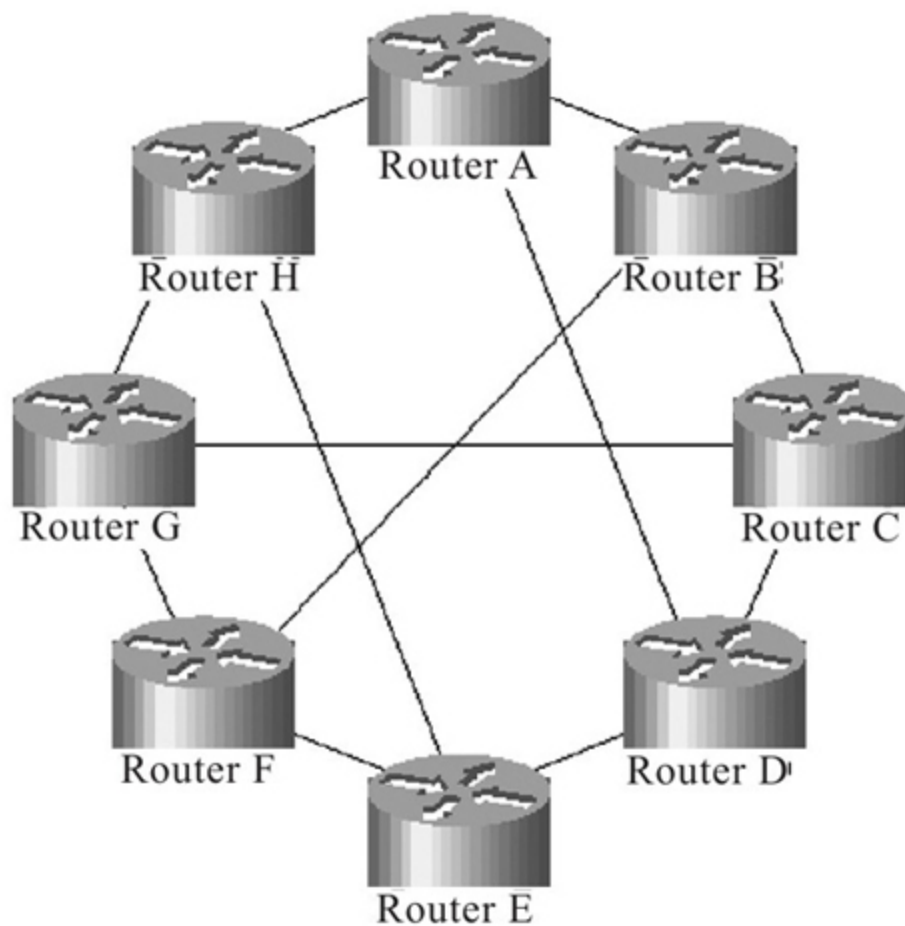


图 2-35 广域骨干网中的半网状拓扑结构示例

网状拓扑结构具有较高的可靠性，因为这种拓扑结构中各节点的连接存在冗余线路，任何单一连接线路中断都不会影响网络的整体连接。同样是由于存在冗余线路，所以比较容易在多条线路上实现负载均衡。但其结构复杂，配置也很复杂，实现起来成本可能很高，特别是在广域网环境，也不易管理、维护和进行网络扩展；同样由于节点间存在多条冗余线路，导致容易出现路由环路，或者二层环路（如果连接的结点是交换机），路由配置复杂。

2.3.7 混合型拓扑结构

混合型网络结构是目前局域网，特别是分布型大中型局域网中应用最广泛的网络拓扑结构，它可以解决单一网络拓扑结构的传输距离和连接用户数扩展的双重限制。

1.混合型拓扑结构概述

混合型网络拓扑结构是指由多种结构（如星型拓扑结构、环形拓扑结构、总线型结构、网状结构）单元组成的拓扑结构，但常见的是由星型拓扑结构和总线型拓扑结构结合在一起组成的，如图2-36所示。

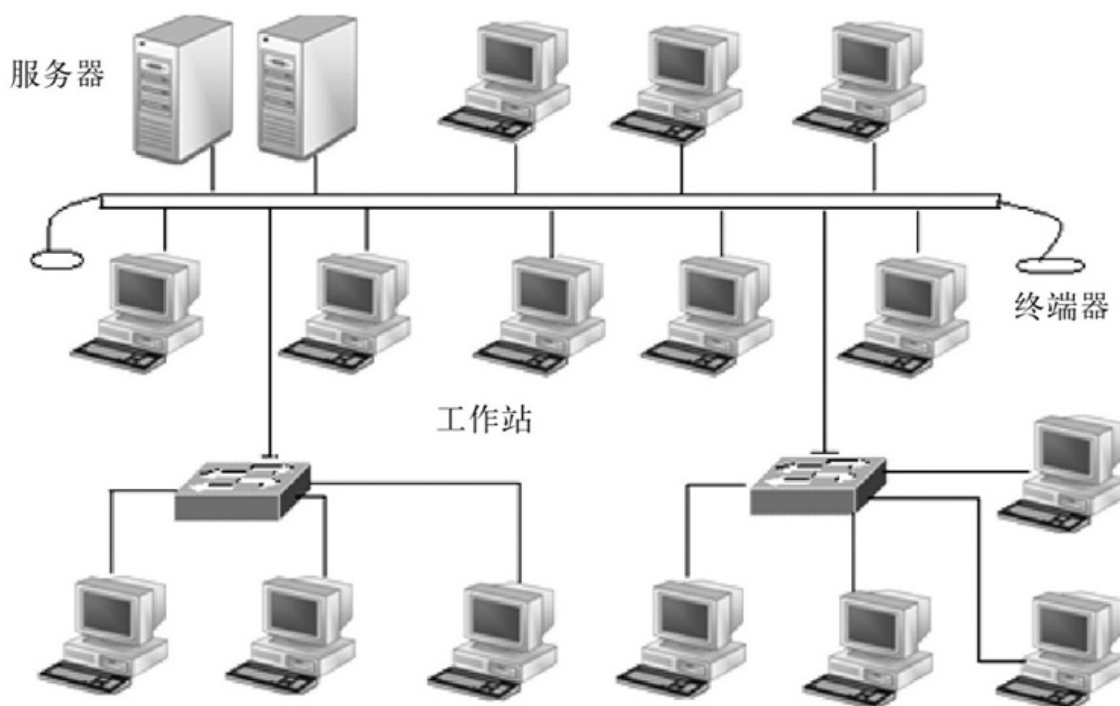


图 2-36 混合型拓扑结构示例

混合型拓扑结构更能满足较大网络的灵活扩展，解决星型网络在传输距离上的局限（因为双绞线的单段最大长度要远小于同轴电缆和光纤），同时又解决了总线型网络在连接用户数量的限制。图2-36所示只是一种简单的混合型网络结构，实际上的混合型拓扑结构主要应用于分布在多层或者多栋建筑物中的网络中。其中采用同轴电缆或光纤的“总线”用于垂直或横向干线，基本上不连接工作站，只是连接各楼层或各建筑物中各核心交换机，而其中的星型拓扑结构网络则体现在各楼层或各建筑物内部的各用户网络中，如图2-37所示。

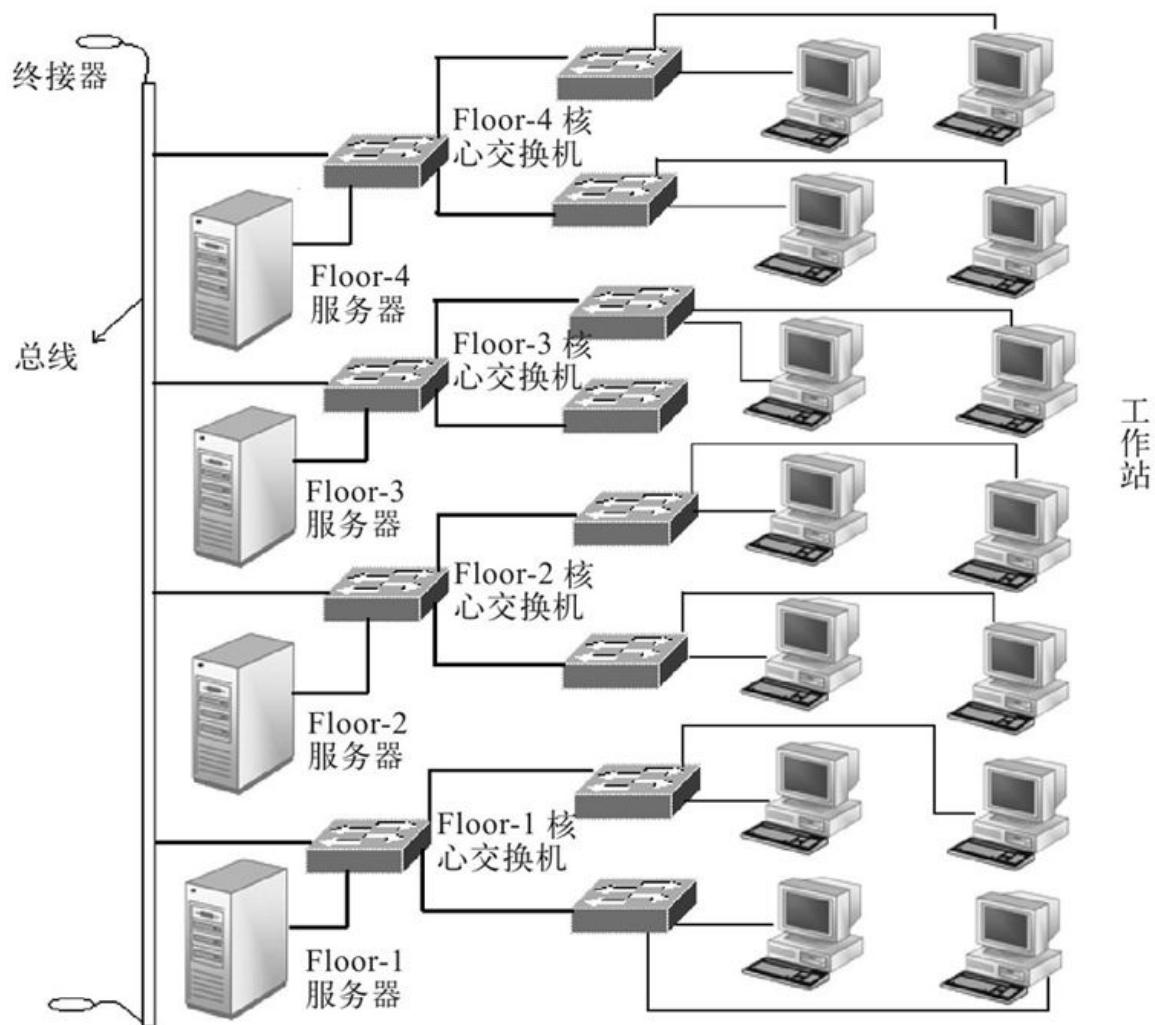


图 2-37 多楼层的混合型网络结构示例

在实际的组网中，现在一般很少使用同轴电缆作为总线了，而是采用传输性能更好，更方便进行网络连接的大对数双绞线。因为在一般的20层以内的楼中，100m的双绞线就可以满足，如图2-38所示。

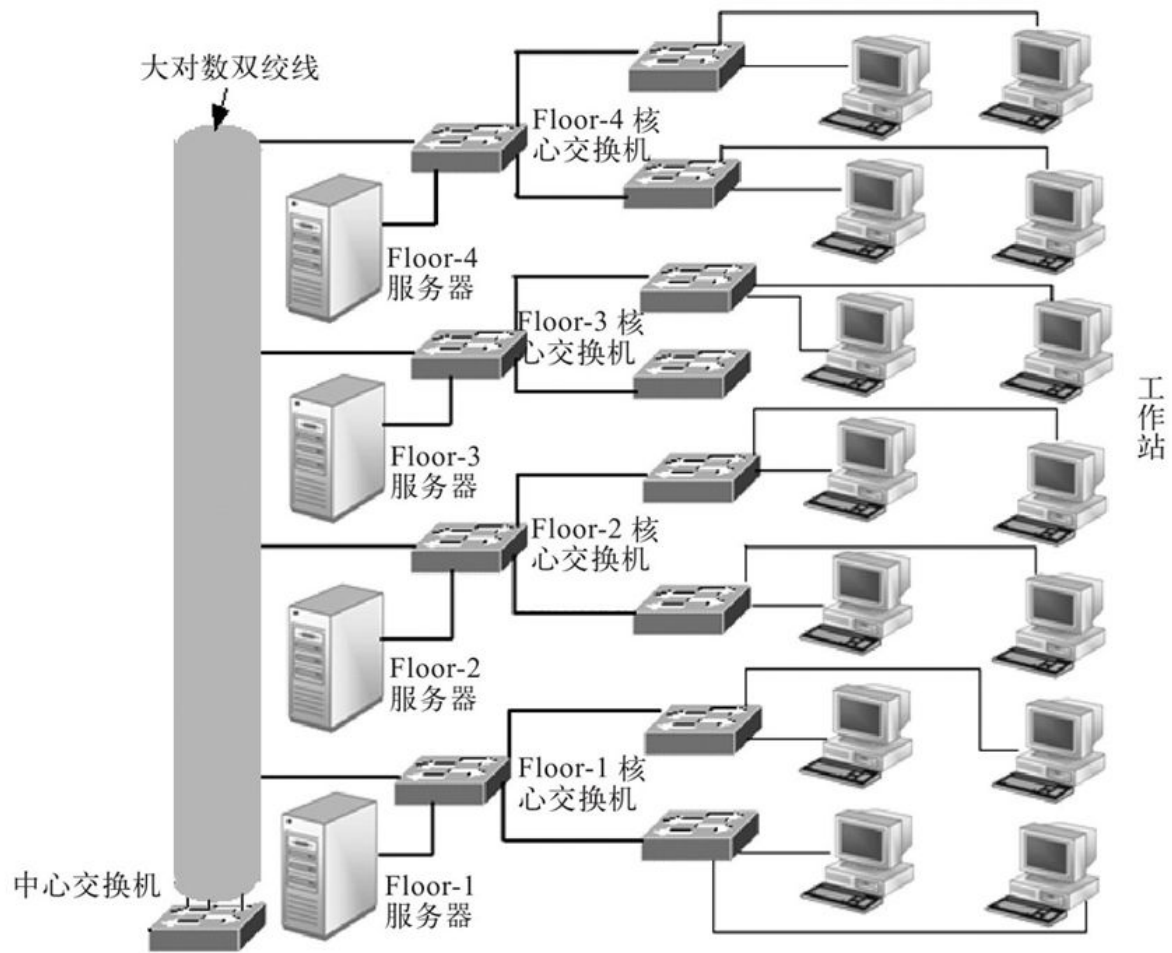


图 2-38 分层星型拓扑结构示例

如果距离过远，如高楼层或多建筑物之间的网络互连，则可以用光纤作为传输介质，无论哪一种，传输性能均要比总线型连接方式好许多。

2.混合型拓扑结构的主要特点

混合型拓扑结构主要有以下几个方面的特点。

(1) 应用广泛

这主要是因混合型拓扑结构解决了星型和总线型拓扑结构的不足，满足了大公司组网的实际需求。目前在一些智能化的信息大厦中的应用非常普遍。在一幢大厦中，各楼层间采用光纤作为总线传输介质，一方面可以保证网络传输距离，另一方面，光纤的传输性能要远好于同轴电缆，在传输性能上也给予了充分保证。当然，这不仅关系到传输介质的选择问题，更重要的是涉及网卡和交换机端口类型的选择，光纤类型端口网卡和交换机要贵许多的。

(2) 扩展灵活

这主要是因为混合型拓扑结构继承了星型拓扑结构的优点。但由于仍采用广播式的消息传送方式，因此在总线长度和节点数量上也会受到限制，不过在局域网中的影响不是很大。

(3) 维护较为困难

这主要受到总线型网络拓扑结构的制约，如果总线中断，则互连的各部分网络也就中断了，特别是对于那些使用统一核心交换机的网络；但是如果是分支网段出了故障，则仍不影响整个网络的正常运行。

2.3.8 无线局域网的两种拓扑结构

在无线局域网（WLAN）中有Ad-Hoc和Infrastructure两种拓扑结构（其实可以理解为WLAN的两种管理模式），前者连接性能较差，连接用户较少（通常为5个以内），主要用于小型家庭和SOHO网络中；后者连接性能较好，主要用于较多用户的企业网络中，应用更为广泛。

1.无线AP的Ad-Hoc模式主要优缺点

Ad-Hoc对等WLAN模式采用的是点对点连接方式（如图2-39所示），只能单点通信，就像有线网络中对等网一样，所以连接性能较差，仅适用于较少数量的计算机无线互连（通常是在5台主机以内）。同时由于这一模式没有中心管理单元，因此这种网络在可管理性和扩展性方面受到一定的限制。而且各无线节点之间只能单点通信，不能实现交换连接。当然这一无线网络结构还是有它自身优点的，那就是网络结构简单，只要安装了无线网卡即可连接，无需其他网络设备，成本非常低。



图 2-39 Ad-Hoc对等无线局域网结构

2.基于无线AP的Infrastructure结构

这种基于无线AP的Infrastructure基础结构模式其实与有线网络中的星型交换模式差不多（如图2-40所示），除了各无线用户需要安装无线网卡外，还需要一个用于集中连接各无线用户的无线AP，它相当于有线网络中的集线器。无线AP都提供了一个有线以太网接口，用于与有线网络、工作站和路由设备的连接。这种网络结构模式主要优势表现在网络易于扩展、便于集中管理、能提供用户身份验证等方面上，另外数据传输性能也明显高于Ad-Hoc对等结构。



图 2-40 Infrastructure基础无线局域网结构

其实图2-40所示也只是一个Infrastructure基础结构单元，在实际的企业WLAN网络中，可能有许多台AP设备，而且还可能有WLAN天线、WLAN中继器、WLAN网桥、WLAN控制器等设备。WLAN天线和WLAN中继器可用于连接更远距离的WLAN用户，WLAN网桥可用于连接不同的WLAN网段，WLAN控制器则可对整个WLAN网络进行管理。这就涉及信道的分配和优化了，在一定范围内不能有信道的重叠，否则信号之间就可能产生冲突。

有关WLAN的主要技术将在第6章具体介绍。

第3章 计算机网络体系结构

从本章起就开始正式介绍计算机网络体系结构（**Computer Network Architecture**）了。本章先从宏观角度介绍各种计算机网络体系结构。着重剖析了它们之间的联系、数据通信原理、各层的数据传输单元、各层数据封装原理以及各层主要功能，本书后面各章将具体介绍这些体系结构中各层主要功能实现原理、主要通信协议以及相关的计算机网络基础知识。

计算机网络体系结构是一个分层次的模块式结构，这样设计的目的的一方面是便于我们从宏观上把握整个网络体系架构，实现快速分析与排除网络故障；另一方面是便于程序开发人员进行网络系统开发时针对不同网络功能进行独立开发，无须考虑其他层的功能。

在计算机网络体系结构中，除了要介绍我们最熟悉的第一个标准化的计算机网络互连体系结构**OSI/RM**（**Open System Interconnection Reference Model**，开放系统互连参考模型）外，还要介绍局域网体系结构**IEEE 802.1**、**TCP/IP**协议体系结构（虽然至今都没有标准化，但已是事实上的标准，是否标准化已没什么关系了）和无线局域网

（**WLAN**）体系结构的**IEEE 802.11**。以上所列的各种协议之间，在存在着一些差异的同时也有着密切的关系，事实上后面几个针对不同计算机网络类型的体系结构都是参照或基于**OSI/RM**来设计或者改进的。

3.1 典型计算机网络体系结构

通过对2.1.2节的学习我们已经知道，早在20世纪60年代末，随着第二代分组交换式计算机网络（杰出代表就是美国的APPANET网络）的诞生，就已出现了计算机网络体系结构的雏形，那就是把整个计算机网络划分为“通信子网”和“资源子网”，如图3-1所示。随后在第三代计算机网络中，ISO推出了第一个标准化的计算机网络体系结构——OSI/RM（Open System Interconnection Reference Model，开放系统互连参考模型）。后面又有一些国际组织或公司先后推出了局域网体系结构IEEE 802.1、TCP/IP协议体系结构（虽然至今都没有标准化，但已是事实上的标准）和无线局域网（WLAN）体系结构的IEEE 802.11。本节先来认识这几种网络体系结构及其一些基本而又非常重要的原理。

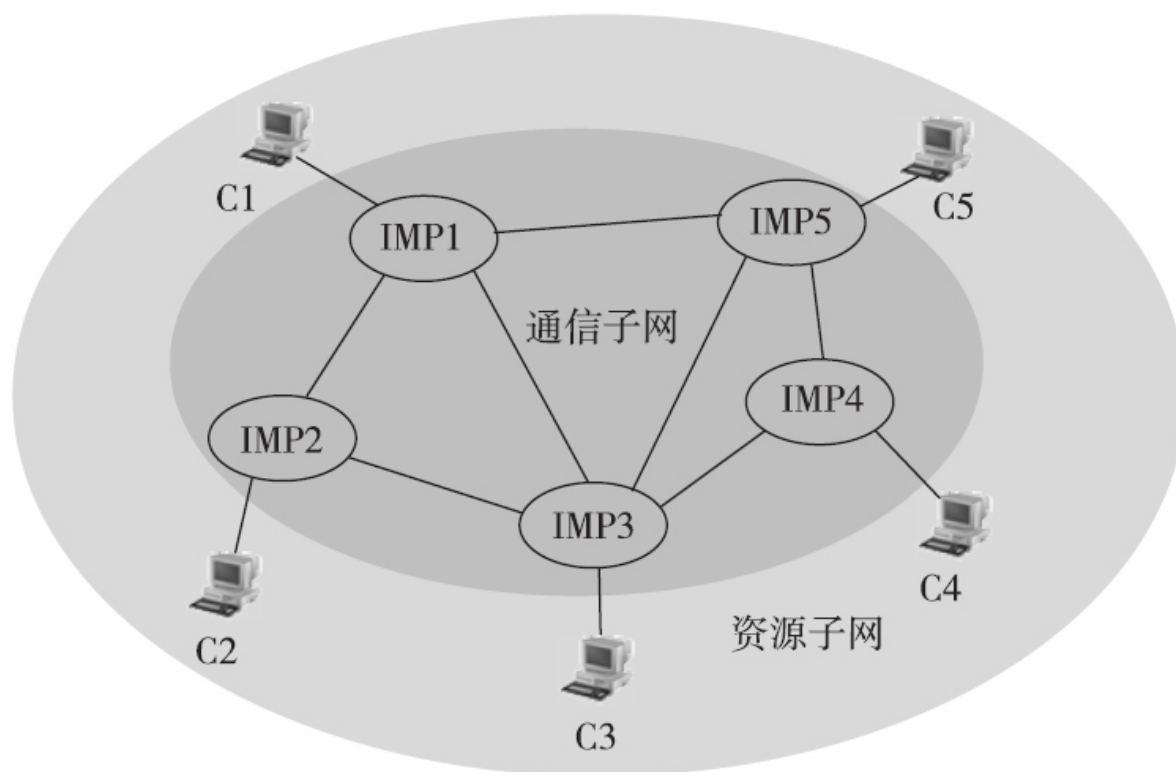


图 3-1 APPANET的基本体系结构

3.1.1 OSI/RM体系结构

OSI/RM体系结构是第一个标准化的计算机网络体系结构。它是针对广域网通信（也就是不同网络之间的通信）进行设计的，将整个网络通信的功能划分为七个层次，由低到高分别是物理层（Physical Layer）、数据链路层（Data Link Layer）、网络层（Network Layer）、传输层（Transport Layer）、会话层（Session Layer）、表示层（Presentation Layer）、应用层（Application Layer），如图3-2所示。但任何广域网其实都是由多个远程局域网连接而成的，所以在

OSI/RM中不仅包括了广域网中不同局域网间通信的功能层次（上面五层），也给出了局域网内部通信所必需的两个层次（最下面两层）。

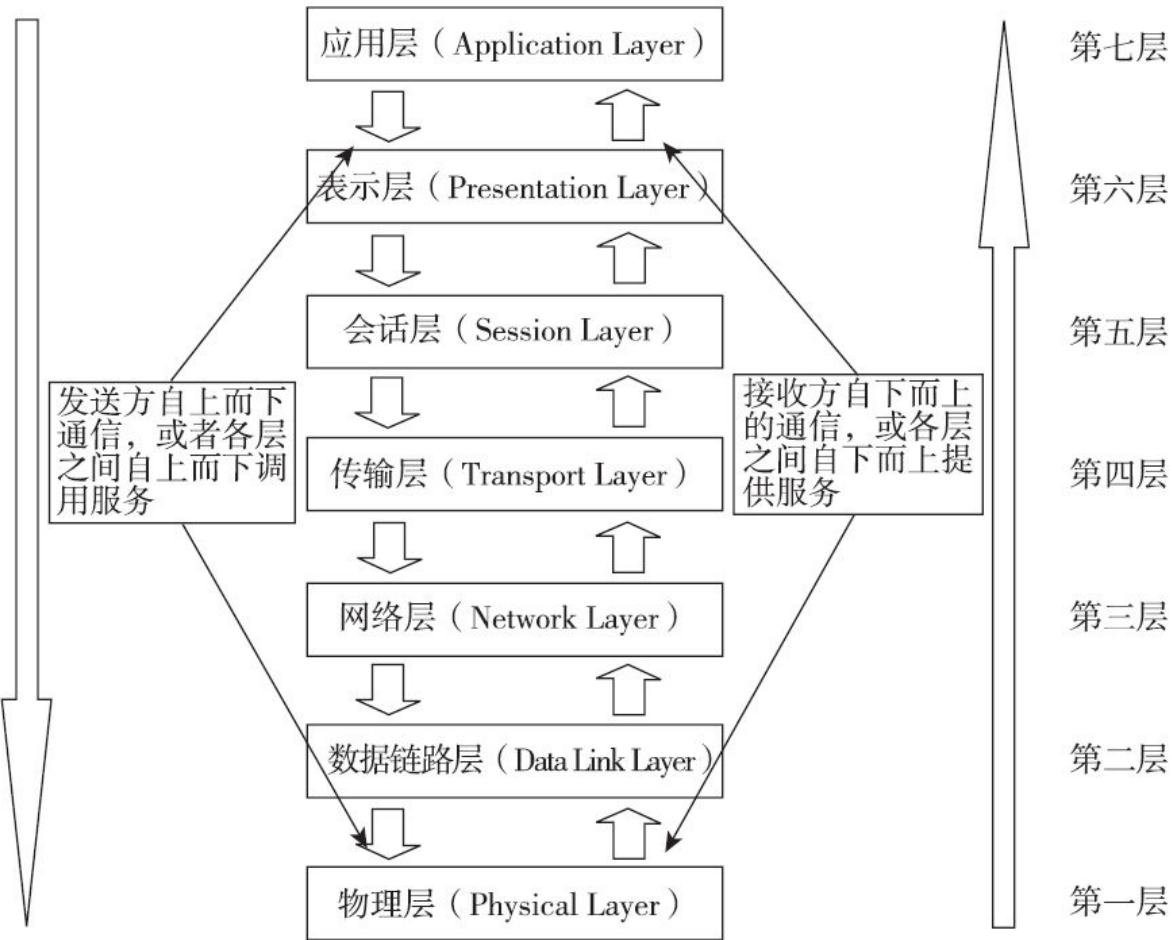


图 3-2 OSI/RM七层参考模型

OSI/RM低四层（从物理层到传输层）定义了如何进行端到端的数据传输，也就是定义了如何通过网卡、物理电缆、交换机和路由器进行数据传输；而高三层（从会话层到应用层）定义了终端系统的应用程序和用户如何彼此通信，也即定义了如何重建从发送方到目的方的应用程序数据流。更多的是把OSI/RM的七层结构分成低三层和高四层

的，低三层负责创建网络通信所需的网络连接（面向网络），属于“通信子网”部分，高四层具体负责端到端的用户数据通信（面向用户），属于“资源子网”部分。OSI/RM结构中各层功能如图3-3所示。

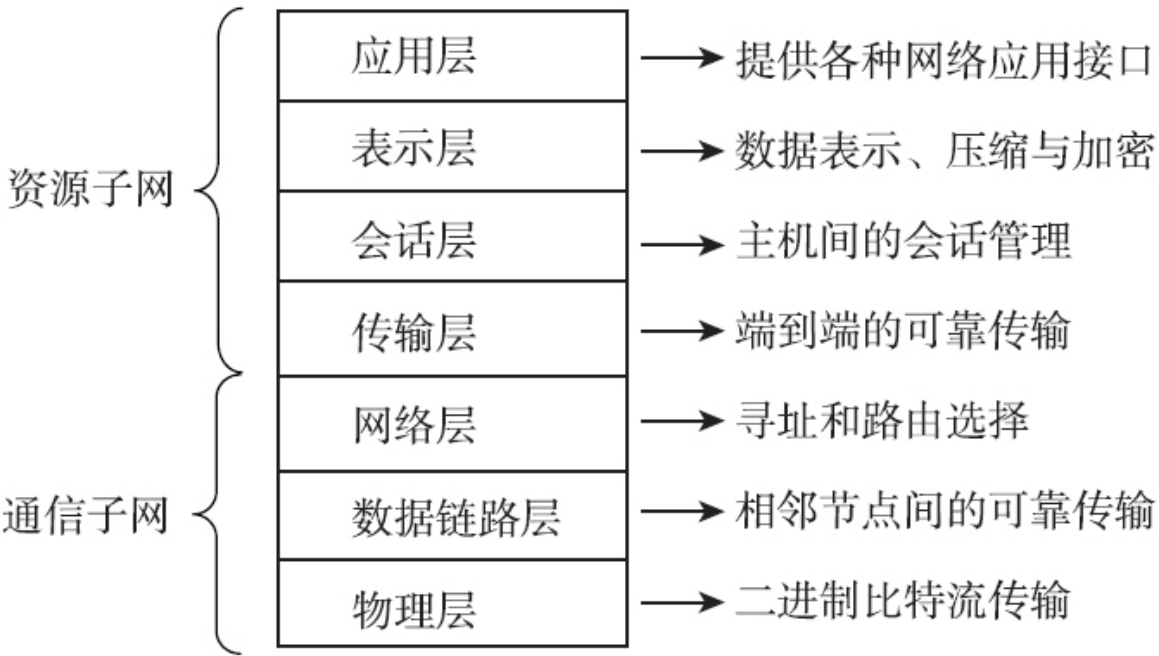


图 3-3 OSI/RM各层基本功能

经验之谈 在OSI/RM中，低三层有两方面的作用：首先是通过它们自己对应层的信息交换构建数据通信所需的网络平台，更通俗地说就是打通一条用于数据传输的网络通道；然后就是为来自上层的数据提供物理的传输通道。但低三层均不能识别和处理来自应用层的网络应用数据，仅用于为用户的网络应用数据通信提供通信线路、网络基础架构，或者说是网络通信平台。高四层上进行的才是真正面向用户

的网络应用，为各种具体的网络应用提供应用平台和端对端的数据传输通道，对低三层所构建的网络平台可以说是“视而不见”。

我们经常听别人说，在局域网中仅可以通过数据链路层的MAC地址进行通信，很多人就认为网络应用也可以仅可以通过数据链路层进行。其实这是完全错误的。这里所说的“通信”其实是仅指两层设备之间的“网络通信”，用于构建数据通信所需的链路。它根本不能识别网络应用的用户数据（这涉及数据的封装次序，具体将在本章后面介绍），怎么可能进行网络应用通信？在局域网中进行具体的网络应用仍需要用到OSI/RM的网络层及以上各层，只是这些高层是由用户主机的操作系统来完成的。

无论是哪种划分方式，OSI/RM的每一层都要完成特定的功能，每层都直接为它的上层提供服务，同时又调用它的下层所提供的服务。所有层次都互相支持，在发送端网络通信是自上而下进行的（也就是自上而下调用服务），在接收端网络通信是自下而上（也就是自下而上提供服务）进行的，但双方必须在对等层次上进行通信（这就是对等通信原理，具体将在本章后面介绍）。当然并不是每一通信都需要经过OSI的全部七层，要视具体通信的类型而定，有的甚至只需要双方对应的某一层即可。如物理层中的物理接口之间的转接，以及中继器与中继器之间的连接就只需在物理层中进行即可。而网络层中的路由器与路由器之间连接则只需经过自网络层以下的三层即可。

经验之谈 无论哪一种计算机网络体系结构，也无论是体系结构中的哪一层，都不是针对具体的设备或者具体的软件而言的，而只是针对每层中所要实现的网络服务功能来划分的。因为每一层所代表的是一组网络功能，而实现某一个功能又可以有许多不同的软/硬件方案。如物理层上就可以有许多不同的传输介质（如同轴电缆、双绞线、光纤等）和网络设备（如集线器、中继器），当然还有许多对应的通信协议。其他各层也一样。计算机网络中的软/硬件是计算机网络通信和数据传输的实体，也就是网络任务的具体执行者。当然各层的实体都不一样，具体将在后面对应章节介绍。

有了这样一个结构模型，就把整个计算机网络软、硬件技术和设备串起起来了，所有软、硬件技术都围绕在这个中心周围。OSI/RM对各个层次的划分遵循下列原则：

□同一层中的各网络节点都有相同的层次结构，具有同样的功能。

□同一节点内相邻层之间通过接口（可以是逻辑接口）进行通信。

□七层结构中的每一层使用下一层提供的服务，并向其上层提供服务。

□不同节点的同等层按照协议实现对等层之间的通信。

□网络设备（不包括计算机主机）间自身的通信仅需要低三层，用来构建数据通信的网络平台。网络平台构建好后，用户应用数据就可以利用这个平台进行各种网络应用通信，但所有网络应用通信都需要经过网络体系结构中的所有层次，其中最上面的四层用来为用户的网络应用通信提供各种服务支持，构建数据通信平台。

但是OSI/RM的七层结构划分从现在看来，并不是很科学，这主要表现在两方面：一是层次数方面还是多了些；二是在进行网络系统设计时仍然觉得比较麻烦。另外，像“会话层”和“表示层”单独划分的意义并不大，因为它们的用途并不像其他层那样明显。所以在后面的TCP/IP协议体系结构中，不再有这两层了。

3.1.2 TCP/IP协议体系结构

TCP/IP协议体系结构（又称TCP/IP协议参考模型）是专门针对使用TCP/IP协议簇的广域计算机网络而开发的，可以说是OSI/RM的改进版本。但绝不能简单地认为是改进版，因为它与OSI/RM所针对的网络类型存在较大区别，所以这两种体系结构中各层所采用的通信协议，以及功能实现原理上都存在非常大的差异。这一点，在后面章节中都会有相应的体现。现在我们常用的通信协议，绝大多数都不是很适用于OSI/RM，而是适用于TCP/IP协议体系结构，因为它们都是应用于TCP/IP网络中。

TCP/IP协议体系结构起源于20世纪60年代末，首先由美国国防部高级研究规划署（Defense Advanced Research Projects Agency，DARPA）作为其研究的一部分，所以又称DARPA参考模型。不仅广域网鼻祖ARPANET使用的是TCP/IP协议体系结构，现在使用最广的Internet也是基于这一模型设计的，因为目前的Internet基本上都是采用TCP/IP协议簇的，包括我们企业内部局域网。

TCP/IP协议体系结构只划分了四层，从高到低分别是：应用层（Appllication Layer）、传输层（Transport Layer）、网际互连层（Internet Layer，又称互联网层）和网络访问层（Network Access Layer，又称网络接入层、网络接口层或者主机-网络层）。虽然只有四

层，但它却包含了OSI/RM中的所有七层的功能，同样包括了局域网和广域网通信所需要的全部功能。图3-4描绘了TCP/IP协议体系结构与OSI/RM层次间的关系。

OSI/RM		TCP/IP协议体系结构
应用层		应用层
表示层		
会话层		
传输层		传输层
网络层		网际互连层
数据链路层		网络访问层
物理层		

图 3-4 TCP/IP协议体系结构及与OSI/RM的比较

从图中可以看出，在TCP/IP协议体系结构中对原来OSI/RM的七层结构进行了进一步的简化，主要体现在以下两个方面：

□把原来的“物理层”和“数据链路层”这两层结构合并为一层，即网络访问层，它提供局域网中的功能；

□合并了原来OSI/RM中的最高的三层，成为新的应用层。因为事实上，在OSI/RM中会话层和表示层的功能都非常单一，完全可以合并到应用层之中。

其他两层，“传输层”与OSI/RM中的功能划分是一样的，而网际互连层也与OSI/RM的网络层实际上是一样的，只不过名称不一样而已。但要注意的是，这里仅是从功能划分上来说的，实际上这两个体系结构是存在相当大差异的。因为OSI/RM是开放型的标准，所以适用于所有类型网络设计参考，而TCP/IP协议体系结构是专门针对TCP/IP网络的，各种通信协议和功能实现原理更加具体。

总体而言，TCP/IP协议体系结构更加精简，更有利于网络系统的设计。但是其中网络访问层本身并不是实际的一层，包括了OSI/RM中的物理层和数据链路层这两层的功能，现在把它们其实合并不是很合理，所以现在通常认为如图3-5（其中与标准的TCP/IP协议四层模型和OSI/RM七层模型进行了对比）所示的五层网络体系结构才是最为科学、合理的。因为它综合了OSI/RM和TCP/IP协议两种体系结构的优点，同时克服了这两种体系结构的不足。本书也将以这种目前广泛建议的五层体系结构进行介绍。

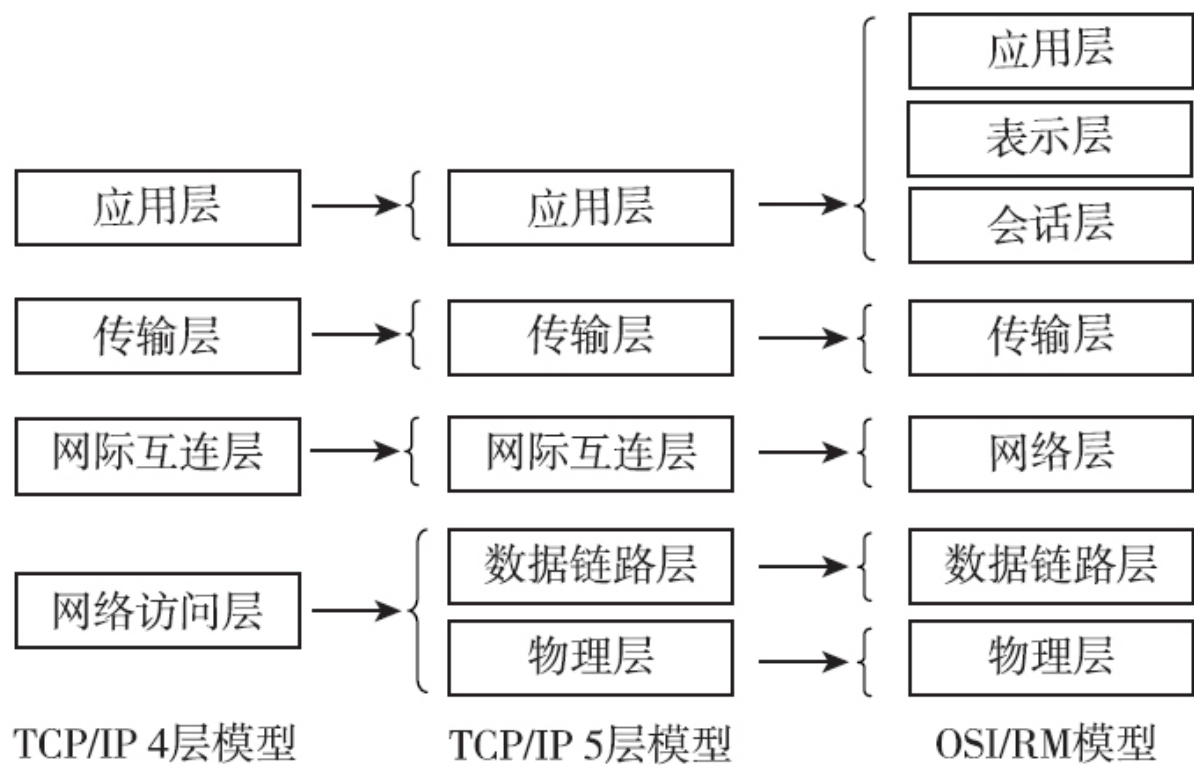


图 3-5 广泛建议的五层网络体系结构

3.1.3 局域网体系结构

目前计算机局域网标准主要是由IEEE发布的，所以局域网体系结构也是由IEEE发布的。它针对有线以太网和WLAN无线局域网分别发布了体系结构，但IEEE也是参考了OSI/RM体系结构来设计局域网体系结构的，可以说是OSI/RM体系结构中专门描述局域网通信的最低两层，或者TCP/IP协议体系结构中专门描述局域网通信的最下面一层，针对以太网所进行的细化。当然其实这两种局域网体系结构从层次上来说是一样的，不同的只是其中各层的功能实现方法和所适用的通信协议不同。

1.有线以太网局域网体系结构

有线以太网局域网体系结构是在IEEE 802.1A标准中定义的，如图3-6所示。从图中可以看出，它仅包括了OSI/RM的最低两层（物理层和数据链路层），因为局域网内部的通信只需要这两层。

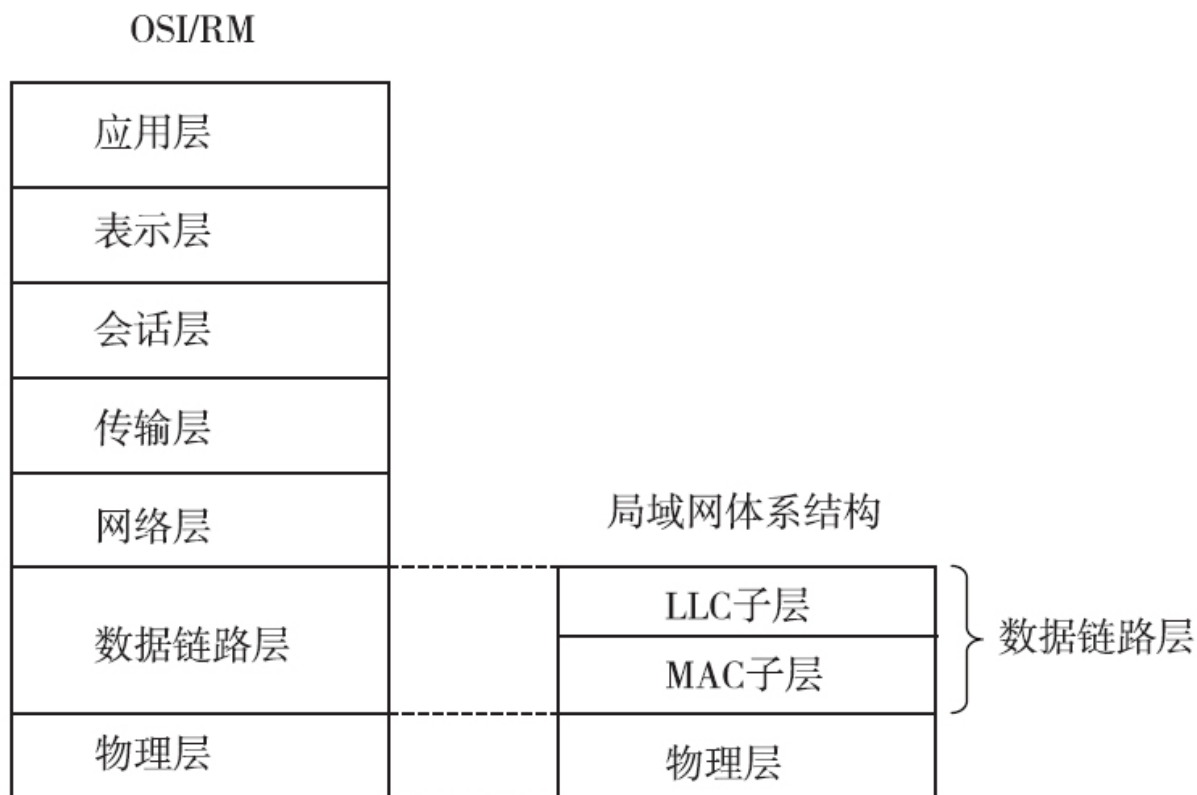


图 3-6 IEEE 802.1A有线局域网体系结构

本节前面介绍的OSI/RM和TCP/IP协议体系结构可以说都是针广域网设计的，并不是局域网通信所全部需要的，这具体表现在两个方面：①在局域网中不需要路由寻址，所以也就不需要它们的网络层，或者说网际互连层；②传输层和应用层的功能是通过安装在计算机操作系统中的网络通信协议和一些具体网络应用软件来实现的，所以在局域网设备中也不需要这两层。

另外，又因为局域网（如以太网）通常是属于广播型网络，存在介质争用现象（广域网中通常是属于点对点网络，通常不存在介质争用），所以它在OSI/RM划分的物理层和数据链路层中又进一步对数据

链路层进行了细分，将其分成了两个子层，即介质访问控制（Media Access Control，MAC）子层和逻辑链路控制（Logical Link Control，LLC）子层。其中的MAC子层就主要是用来解决介质争用和局域网内部寻址的。

IEEE 802.1A局域网体系结构中的物理层与OSI/RM和TCP/IP协议体系结构中的物理层功能是一样的，但仅支持IEEE 802系列局域网标准（如IEEE 802.3系列、IEEE 802.4、IEEE 802.5）中的物理层协议；MAC子层则主要是支持IEEE 802系列局域网标准中的载波监听多路访问（Carrier Sense Multiple Access，CSMA）、载波监听多路访问/冲突检测（Carrier Sense Multiple Access/Collision Detect，CSMA/CD）协议，以及把物理层的比特流封装成MAC子帧的功能；LLC子层主要支持IEEE 802系列局域网标准中的LLC子帧封装、链路控制和管理功能。这些都将在第6章具体介绍，在此不再赘述。

2.WLAN体系结构

WLAN也是局域网的一种，所以WLAN体系结构与上面介绍的IEEE 802.1A局域网体系结构是完全一样的（如图3-6所示），只不过因为两种局域网所用的物理层传输介质，以及相关数据链路层技术或协议不一样，所以在WLAN体系结构中的这两层中所包括的内容也不一样，具体如图3-7所示。有关的具体标准和技术将在第6章介绍。

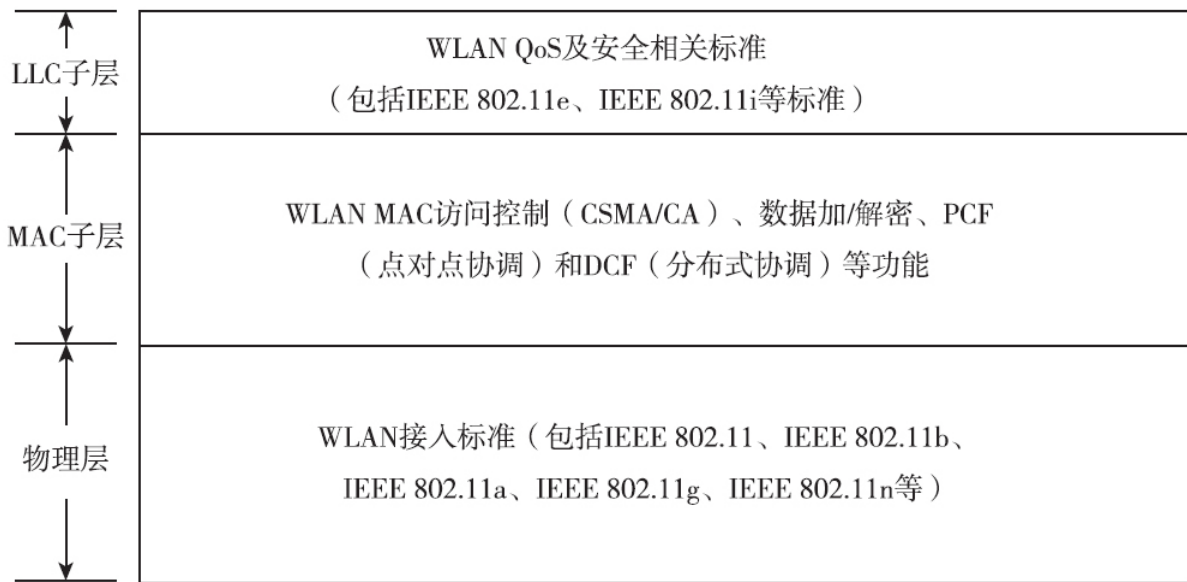


图 3-7 WLAN体系结构中所支持的协议和标准

3.1.4 例说网络体系结构各层主要功能

本节将以最常见的生活事例来诠释OSI/RM计算机网络体系结构中各层的基本功能（其他体系结构对应层次的功能类似，参照即可），以便能使读者更加容易理解。当然或许里面有些比喻并不是很贴切，但只要对理解这些层次的功能有所帮助就达到目的了。

1.物理层

物理层是OSI/RM以及其他所有计算机网络体系结构的最底层，为所有网络/数据通信提供物理的通信线路。物理层是用来构建计算机网络通信和数据传输的通道，相当于日常交通网络中的各种道路，如公路、铁路和航线，它们是我们出门旅行必须要依靠的基础设施。但物理层不是针对具体的传输介质、设备和通信协议的，因为它们可以有多种选择（如传输介质中就可以有同轴电缆、双绞线和光纤等），只要能实现物理层的某种功能就行了。

不同的传输介质和设备选择，必须要有对应的通信协议支持，而且这也决定了不同的选择有不同的物理层性能。就像路有好多种一样，如有泥巴路、沙子路、水泥路、柏油马路、普通铁路、高速铁路等，这些不同的路可以承载的重量和速率都不一样。不同的路相连就

形成了我们旅行途经的整条路径，同样，计算机网络中的不同物理层相连也构成了双方通信的整条路径。

另外，我们知道在各种道路中都会划分许多车道的，在计算机网络体系结构中也有类似的“车道”，那就是我们通常所说的“信道”。信道的全称就是“信号传输通道”，默认情况下，一条物理线路就是一条信道，但也可以通过各种信道复用方式在一条物理线路中划分出多条信道。有关信道复用方式将在第4章介绍。

2.数据链路层

数据链路层为同一局域网内部的网络/数据通信提供点对点的数据传输通道，通过MAC地址寻址把数据转到目的节点，可以理解为我们的市内公路+交通法规。之所以只能理解为市内公路，是因为在各个网络中的数据链路层间的通信仅可以在同一网段内进行；之所以还要加上“交通法规”，是因为数据链路层所提供的不再是物理线路，而是在物理层的物理线路基础之上，通过数据链路层协议（相当于市内交通法规）构建的，可真正用于数据传输的虚拟数据传输通道，但这样的虚拟数据传输通道也只能在同一网段内进行数据转发。数据链路层仅为所到达数据在本网段内进行转发提供传输通道，要在不同网段间进行数据转发，还必须依靠下面将要介绍的网络层和传输层。

注意 链路可分为物理链路和逻辑链路。物理链路可以看成是在物理层中相邻结点间的那段线路，而数据链路则在物理链路基础上再封装上对应的数据链路层通信协议，是可以实现数据传输的逻辑链路。

3.网络层或网际互联层

OSI/RM中的网络层（或TCP/IP协议体系结构中的“网际互连层”）为不同网段之间的数据转发提供路径选择，通过IP地址（也可以是其他网络层地址，要视具体网络类型而定）把数据包转发到目的节点，可以理解为交通网络中的车站、机场、码头。这涉及一个选择下一站路径的问题，也就相当于我们要到某外地去旅行，到了车站、机场、码头后要选择乘坐哪趟车、哪趟飞机或轮船才能最快捷，成本最低。

网络层的这种寻址功能就是我们通常所说的“路由寻址”，就是选择哪条路径来到达下一个路由结点。通过不同的路径进行路由，在性能、成本上都可能不一样，就像我们选择不同的交通工具，或者不同线路旅行时的效率和成本都不同一样。网络层的功能就相当于连接不同网络的桥梁，仅起到在不同网络间转发数据包的作用，最终数据还是要在目的网络的数据链路层进行传输，在到达下一个网络结点设备（如路由器）时再进行路由、转发。

另外，就像我们到达另外一个城市又得遵照另外城市的交通法规一样，当我们的通信包到达另一个网络时，同时要遵照这个网络中的链路规则，也需要有相应的链路层协议来支持，以最终完成数据的传输。

4.传输层

“传输层”是在下面三层构建的网络平台基础上专门为通信双方构建端对端（不是点对点）的数据传输通道，使通信双方就像直接进行数据传输一样。这个端对端传输通道是可以跨网络的，这与数据链路层所构建的仅用于局域网内部的点对点传输通道是不同的。

传输层类似于国际航线，一条国际航线可能要经过几个国家，但国际航班飞机飞行时根本不用考虑经过了哪些国家的航线，因为这些事先在确立国际航线时就已处理好了，就像一条传输通道要经过几个网络不用管一样，因为网络层事先已准备好了通信路径。

5.会话层和表示层

这两层仅在OSI/RM中单独划分，而在TCP/IP协议体系结构中是没有这两层的，那是因为TCP/IP协议是专门针对TCP/IP协议类型网络而开发的体系结构，不存在其他网络类型，所以不需要表示层，会话层的作用因为太单一，所以合并到了应用层中。

会话层为具体的用户应用建立会话进程（每个应用都有一个会话进程），这个过程是一个用户网络应用的协商过程，相当于车站、机场或码头中总调度人员所从事的调度工作。

表示层是对用户网络应用数据的具体解释，包括在网络通信时可采用的信息格式、可采用的加密方式，相当于车站、机场、码头中发送每一班次汽车、火车、轮船的具体文件，包括所采用的车型、机型、船型，以及所负责的运输公司、交接人员名单等内容。

6.应用层

应用层是用户进行具体网络应用的层次，是具体网络应用的体现者。应用层负责接受用户的各种网络应用进程的调用，相当于车站、机场和码头的负责人负责接受乘客运输的调度，确定具体班次的发送时间和要完成的任务。负责人一声令下，下面的所有相关工作人员都得围绕他的指令进行准备。应用层也一样，只要网络用户有需要，通过相应的网络应用软件就可以发出相应的指令，然后通过应用层相关的通信协议来接收，并向它的下面各层依次传达并使其执行具体的网络应用指令，进而完成整个网络应用任务。

OSI/RM七层结构各层的主要功能如表3-1所示。

表 3-1 OSI/RM 各层主要功能

层 次	主 要 功 能
物理层	<ul style="list-style-type: none"> • 规定网络设备的机械特性和电气特性，为网络 / 数据通信提供物理连接和传输通道 • 为数据信号进行编码，提供比特流的透明传输
数据链路层	<ul style="list-style-type: none"> • 建立网络和数据通信的逻辑传输通道，使有差错的物理线路变成无差错的数据链路 • 为同一网络内部通信提供两层 MAC 地址寻址及帧格式封装 • 以帧为基本格式对数据提供流量控制和差错控制
网络层	<ul style="list-style-type: none"> • 为不同网络间的主机通信提供网络寻址和路由转发 • 以分组为基本格式提供流量控制、拥塞控制和差错控制
传输层	<ul style="list-style-type: none"> • 以端到端方式建立数据传输连接和通道，屏蔽途经网络中所有低层服务上的差异 • 以数据段为基本格式提供流量控制、拥塞控制和差错控制
会话层	<ul style="list-style-type: none"> • 维护通信双方应用进程会话 • 管理通信双方数据交换进程
表示层	<ul style="list-style-type: none"> • 数据格式转换 • 数据加密与解密 • 数据压缩与解压缩
应用层	为各种网络应用提供服务

3.1.5 OSI/RM和TCP/IP协议体系结构的比较

TCP/IP协议体系结构是在OSI/RM基础上，专门针对TCP/IP网络而开发的体系结构，所以它既有OSI/RM的基本模型结构和层次划分思想，又针对了特定的TCP/IP网络，所以其更加具体化，更加具有可操作性。本节要具体介绍这两种体系结构的主要异同。

1.相同之处

总体而言，OSI/RM和TCP/IP协议体系结构主要具有以下几个方面的相同或相似点：

(1) 层次结构划分思想相同

这两种体系结构都是以协议栈（不同协议形成的层次结构）为基础进行层次结构划分的，并且协议栈中的协议是彼此独立的。这样做的好处是，可以大大简化各种网络协议程序的设计，只需要为不同协议程序提供关联的程序接口即可。

(2) 总体层次结构相似

在这两个体系结构中，虽然总的层数和对应层次名称都有所不同，但总体层次结构还是极其相似的。TCP/IP协议体系结构中的“网络

访问层”对应了OSI/RM最低的“数据链路层”和“物理层”这两层，TCP/IP协议体系结构中的“应用层”对应了OSI/RM最高的“会话层”、“表示层”和“应用层”这三层，OSI/RM中间的“网络层”虽然与TCP/IP协议体系结构中的“网际互连层”在名称上不一样，但功能却是完全一样的，至于“传输层”，则两种结构都是完全一样的。而且在这两种结构中，“传输层”以下都属于“通信子网”部分，用来构建通信网络；而“传输层”及以上各层都提供了端到端的、与网络无关的服务，属于“资源子网”部分。

(3) 核心组成一样

这两种体系结构中都定义了“服务”、“接口”和“协议”三个重要核心。

“服务”也就是后面各章所说的“服务原语”，定义了各层应该做些什么，要提供哪些功能，但没有定义本层该如何工作，以及上一层该如何访问本层。

“接口”也就是后面各章所说的SAP（服务访问点），为对应的上层提供了获取本层服务的逻辑接口，规定了有哪些参数可以使用，以及使用这些参数的结果是什么。

“协议”也就是标准中所说的“通信规程”，它是各层服务功能的具体实现者。当然同一服务在不同网络中可以用不同的协议来实现。各

层中的协议各自实现自己的功能，并不影响其他层，任何的一层都只为相邻的上一层提供服务。

2.不同之处

在看到OSI/RM和TCP/IP协议体系结构的相似之处的同时，也要看到这两种体系结构的许多不同之处。两者的不同之处主要表现在：

(1) 适用范围不同

OSI/RM在标准化协议发明之前就已产生了，所以OSI/RM不偏重于任何特定的网络类型，具有最广泛的理论上的参考性，是一个理想化的模型。而TCP/IP协议体系结构则相反，它是在TCP/IP协议簇先出来了（而且仅适用于TCP/IP类型网络）后，再针对这些协议进行功能分层和描述的，所以与协议的关系非常紧密，两者的吻合得非常好，最具实践性。

(2) 层次结构不同

这两种网络体系结构在层次划分上的不同主要体现在：TCP/IP协议体系结构中没有“会话层”和“表示层”，因为事实已证明这两个层次并没有多大用途，即使在OSI/RM中也一样，所以最后取消了，它们的功能合并到“应用层”之中。

另外，OSI/RM中的“物理层”和“数据链路层”的功能在TCP/IP协议体系结构中合并到了“网络访问层”中，尽管实际上在TCP/IP协议体系结构中对这个层中的具体功能并没有明确规定，但实际上这层功能就是OSI/RM最低的这两层的功能。当然，这也是大家普遍认为TCP/IP协议体系结构层次划分中不科学的一个重要方面。

(3) 支持的网络通信模式不同

OSI/RM的网络层同时支持无连接和面向连接的网络通信（它不仅支持TCP/IP协议网络中无连接的IP网络协议，同时支持NetWare SPX/IPX网络中的面向连接的SPX服务等）；TCP/IP模型的网络层只提供无连接的服务（因为它只支持IP这种无连接的网络层协议）。

(4) 所包括的通信协议不同

OSI/RM是一种开放型的，希望尽可能适用于所有类型计算机网络的理想化体系结构模型，所以它里面所包括的通信协议不仅非常多，且类型非常复杂，适用于各类网络的都有。但是，由于现在网络系统设计者通常不是参考OSI/RM，而是以目前占据了绝大部分市场的TCP/IP体系结构作为体系结构设计参考，所以现在OSI/RM中的许多通信协议都已过时。尽管TCP/IP网络也在OSI/RM的设计范围内，但现在TCP/IP网络中的通信协议是专门针对具体的TCP/IP协议体系结构而开

发的，所以更具有**TCP/IP**协议体系结构的特点，而且这些通信协议在不断改进，非常适用于目前广泛应用的**TCP/IP**网络。

3.2 计算机网络体系结构通信原理

无论是以上哪种计算机网络体系结构，在通信原理上它们都有着两个相同的方面：一个是在网络连接和数据传输流程方面，发送端是自上而下（也就是从高层到低层），接收端是自下而上（也就是从低层到高层）进行的，也就是数据通信原理相同。另一个是通信会话方面，双方都必须是逻辑上的对等层次，也就是对等通信原理相同。如发送端的网络层只能与接收端的网络层通信，而不能直接与数据链路层或者与传输层进行交错通信。其他的层次也是一样的。下面分别予以介绍。

3.2.1 网络体系结构的数据通信原理

在各种计算机网络体系结构的网络连接建立和数据传输的流程中，发送端是把通信连接建立指令和用户应用数据从上层向下层传输的，直到最低的物理层；而接收端是把通信连接建立指令和用户应用数据从下层（从最低的物理层开始）向上层传输的，直到与发送端发起通信的对等层。图3-8所示的是OSI/RM情形下数据的传输流程，其他体系结构的数据传输流程与此类似，只是数据传输中流经的层次上有些区别。

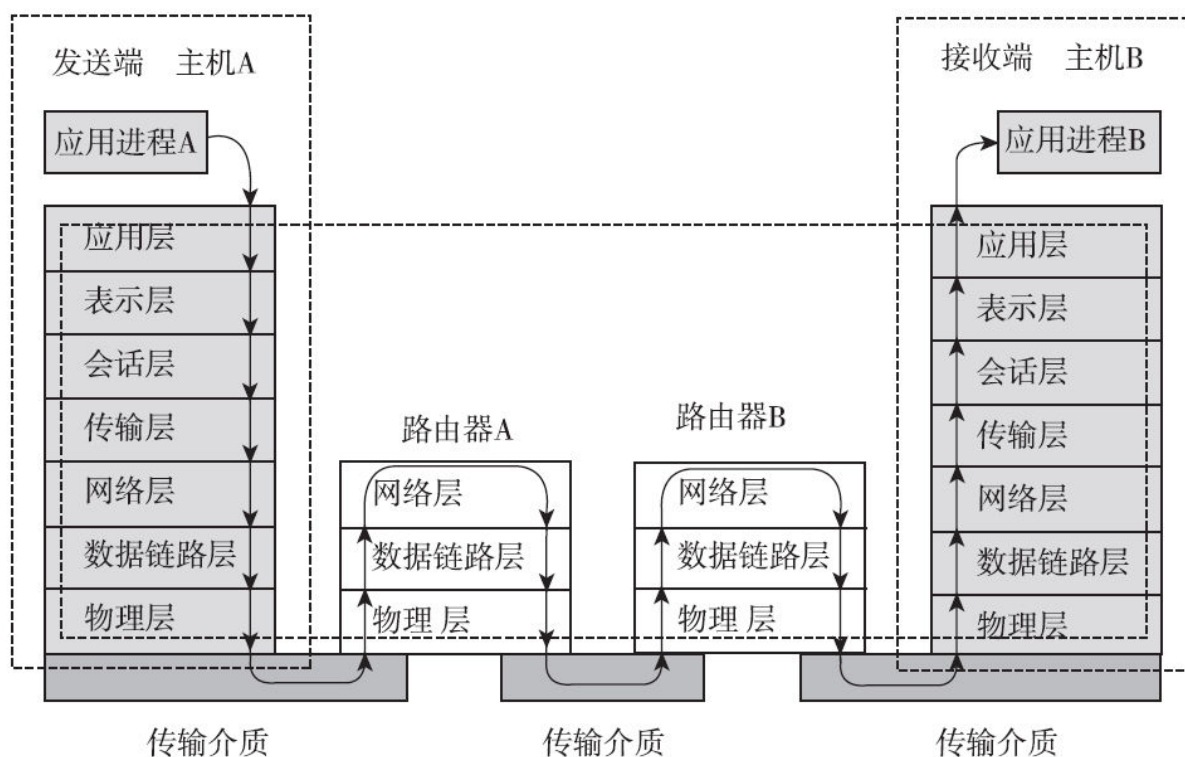


图 3-8 OSI/RM数据通信原理示意图

其实这也很好理解，毕竟网络通信连接和数据传输都不能凭空建立或进行。它必须通过计算机网络中最基础、也是网络体系结构中的最低层——物理层，通过传输介质来传递各种通信连接建立信号和数据。但在进行具体的数据传输前，必须先建立好相应的连接。建立的连接可以是永久连接（如局域网中的连接和广域网中的各种专线连接），也可以是非永久连接（如各种拨号连接和串行线路）。就像我们上互联网都要拨号连接（专线连接的除外），打电话我们首先也要拨号一样。再如，当我们要运输货物到外地，首先得找好运输货物的路径，联系好接收货物的人，在对方确认可以收货后才可以发货，否则路都不知道怎么走，对方都不想要这批货物，怎么能发货呢？

计算机网络体系结构中的这种源端自上而下，目的端自下而上的通信连接建立和数据传输流程，与我们在公司中安排、完成一个具体任务的流程是一样的。

现假设A公司的总经理要与B公司的总经理签个协议，一般是按照如下流程进行的：

- 1) A公司总经理把这个要求向他的某下级部门经理交代；
- 2) 该部门经理又会把这项具体的任务交代他下面的某个负责这方面工作的员工，让其做好相应准备；
- 3) 具体负责的员工与对方公司取得联系。

B公司的执行流程与A公司的正好相反，因为这个任务请求最先是由B公司下面负责具体联系工作的员工收到的。在他接收到A公司的这个请求后，会向他的直接上级部门经理反映，再后这个部门经理又会把这个请求向B公司总经理反映。B公司的总经理还可能要根据具体的工作任务安排具体的签协议日程，然后把这些信息依次向下反馈到公司中具体负责这方面的工作人员，再与A公司的具体工作人员沟通，A公司人员接到B公司的这些信息后，又要依次向上传达到A公司的总经理。在这个流程中可能要经过多次反复，最终完成A公司总经理与B公司总经理整个签协议事件的信息下达（在A公司）、上传（在B公司）

任务。接下来的事，就是两家公司总经理之间的事了，剩下的事就相当于本节后面要介绍的通信会话原理。

3.2.2 网络体系结构的对等通信原理

通信双方的网络连接建立好后，就可以进行各种具体的网络应用和网络通信了，但这时的通信是建立在双方对等层次上的，也就是我们通常所说的“对等层”（peer layers）通信原理。进行对等通信主要出于两方面的考虑：一方面是因为只有双方是对等层次的会话才可能使用相同类型的协议，彼此才能“听得懂”，才能有“共同语言”；另一方面是因为在网络体系结构中，每一层都是独立完成自己工作的，其他层都是不干预，不了解的。如一方的物理层只能与对方的物理层直接通信，不可能直接跳到与对方的“数据链路层”或其他层进行对话。同理，一方的网络层只能与对方的网络层通信，一方的传输层只能与对方的传输层通信，以此类推。只有“物理层”之间的会话才是直接的，其他各层之间的会话都是逻辑意义上的，如图3-9所示（注意其中的虚线）。

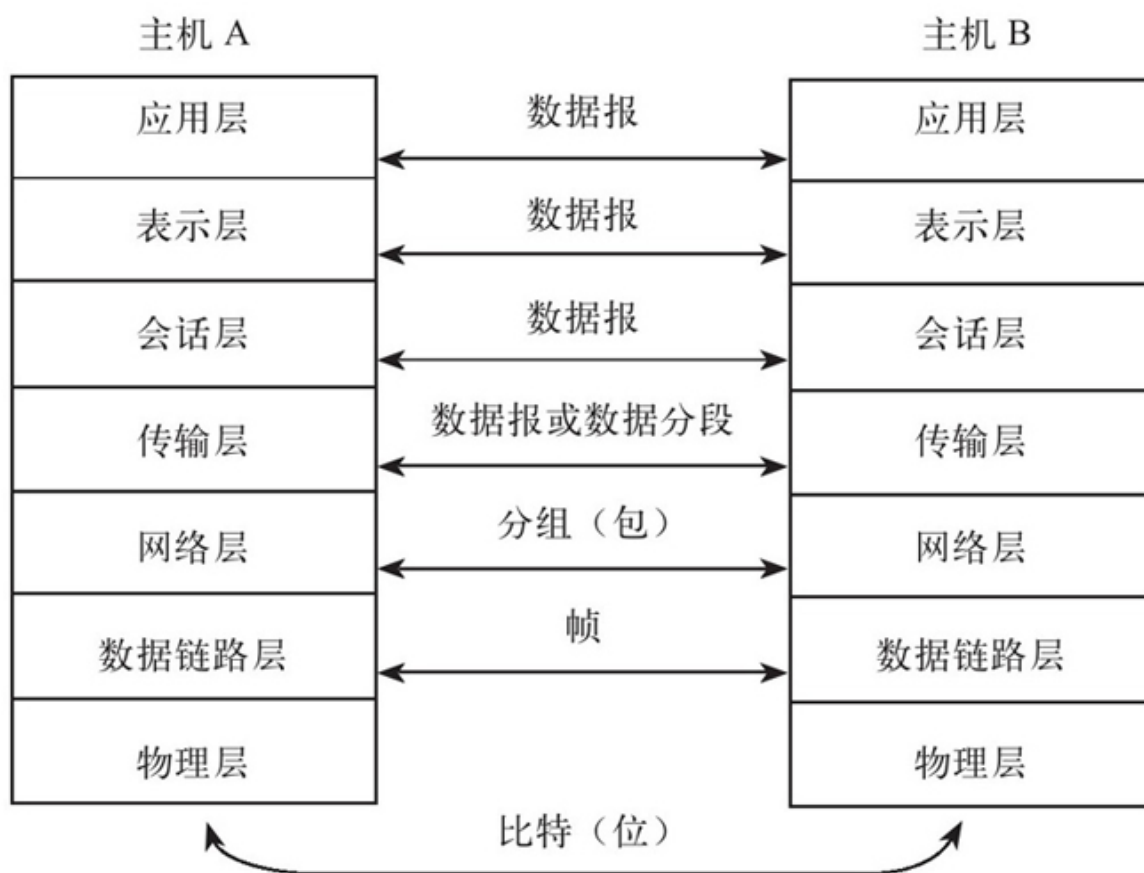


图 3-9 OSI/RM对等通信原理示意图

注意 尽管必须遵守对等通信原理，但最终的数据通信流程还是要经过发起会话通信的层次的下面所的其他层次的，只不过，在具体的通信过程中，对于用户来说，下面各层是透明的，在网络连接建立好后，我们并不需要了解下面各层的工作情况。所以我们在进行具体的网络应用（如文件传输）时，似乎是直接把文件传给了对方。

(1) 各层上传输的数据格式

对等层之间传送的数据单位称为PDU（Protocol Data Unit，协议数据单元）。不同层的PDU所包括的内容和格式也不一样。这就要涉及各层传输的数据格式问题了。

在各种网络体系结构中的每一层传输的数据格式，或者说协议数据单元（PDU）是不一样的，具体如下：

□“物理层”是以最原始的“比特”（bit）流格式传输的，或者说物理层的PDU就是“比特”。

□“数据链路层”的传输单位是“帧”（frame），一个帧包括多个比特，但一个帧的大小必须是一个整数字节。不同协议的帧大小也不一样。一个帧其实也就是一个DPDU（数据链路协议数据单元）。

□“网络层”的传输单位是“分组”（或者“包”，paket），一个分组又可以包括多个帧，分组大小也要根据不同协议而定，一个分组其实也就是一个NPDU（网络协议数据单元）；

□传输层比较特殊，OSI/RM体系结构中是直接以TPDU（传输协议数据单元）为单位的，而在TPC/IP协议体系结构中，TCP是以数据段（segment）为单位进行传输的，UDP是以数据报（datagram）为单位进行传输的。

□在会话层、表示层和应用层中是以具体的数据报文为单位进行传输的。

以上各层的数据传输单位如图3-9所示。如果是其他网络体系结构（如TCP/IP协议体系结构、局域网体系结构），各层上传输的数据格式也是一样的，不同的只是少了一些层次而已。

（2）协议头和协议尾的封装

在整个数据传输过程中，数据在发送端时经过各层时都要附加上相应层的协议头和协议尾（仅数据链路层需要封装“协议尾”）部分，也就是要对数据进行协议封装，以标识对应层所用的通信协议。

“协议头”是用来封装本层PDU的，“协议尾”则代表本层封装的结束。如在我们常见的以太局域网中传输的帧都会封装对应的数据链路层——以太网协议，其中就包括MAC子层协议头和LLC子层协议头，具体将在第6章介绍。图3-10中左边箭头所示的顺序就是OSI/RM各层的数据封装流程，其中的AH为应用层协议头，PH为表示层协议头，SH为会话协议头，TH为传输层协议头，NH为网络层协议头，DH为数据链路层协议头（物理层为最低层，传输的是最小单位的bit（比特），不需要再进行封装，所以没有“物理层头”），DT为数据链路层协议尾。

在数据的接收端，数据是由低层向高层传输的，这样当数据到达某一层后，就会去掉对应下层的协议头和协议尾部分，这个过程就是一个解封装的过程，是前面协议封装的逆过程，如图3-10中右边箭头所示的顺序。因为上层并不需要了解它的下层服务，所以当包或帧送到某一层时就会把用来标识它下一层的协议头和协议尾去掉，还原该包或帧在发送端对应层时的包或帧内容。其实加上协议头的作用可以理解为在发送端要一层层地加上一个指明到达下层地址的信封，而在接收端则要一层层地拆开信封，以获取向上层传输的地址信息，使数据能继续向上层传输。

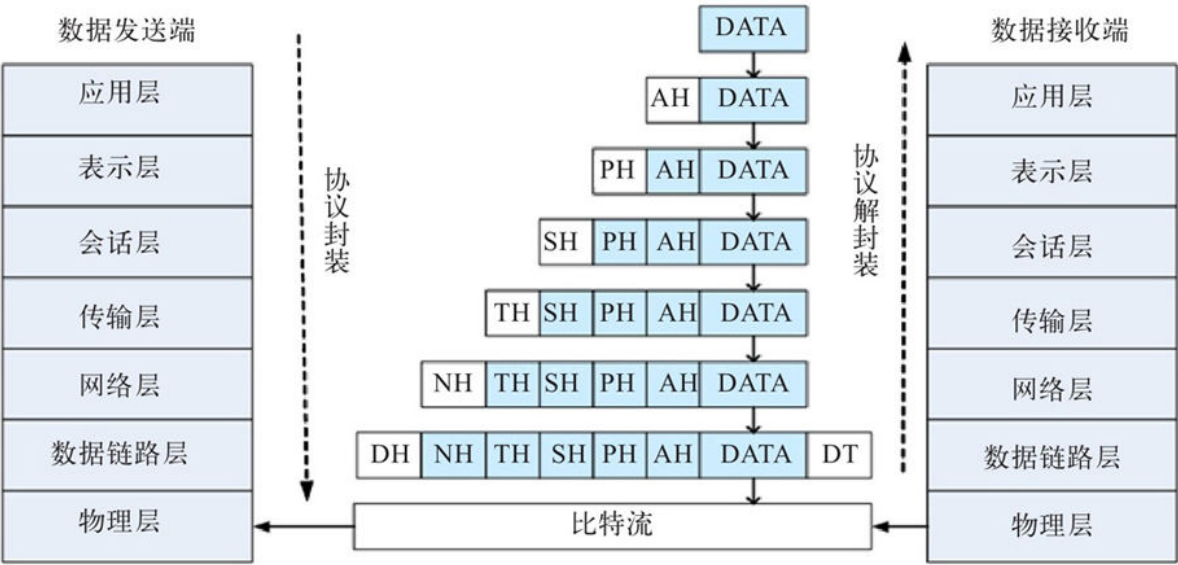


图 3-10 OSI/RM各层数据封装流程

TCP/IP体系结构中的数据封装和通信原理与OSI/RM类似，只不过没有OSI/RM这么多层次。图3-11所示的是两个中间隔了两个网段的主

机（Ha和Hb），在TCP/IP网络中的基本通信流程（注意箭头的方向，假设通信是由Ha主机发起的）。

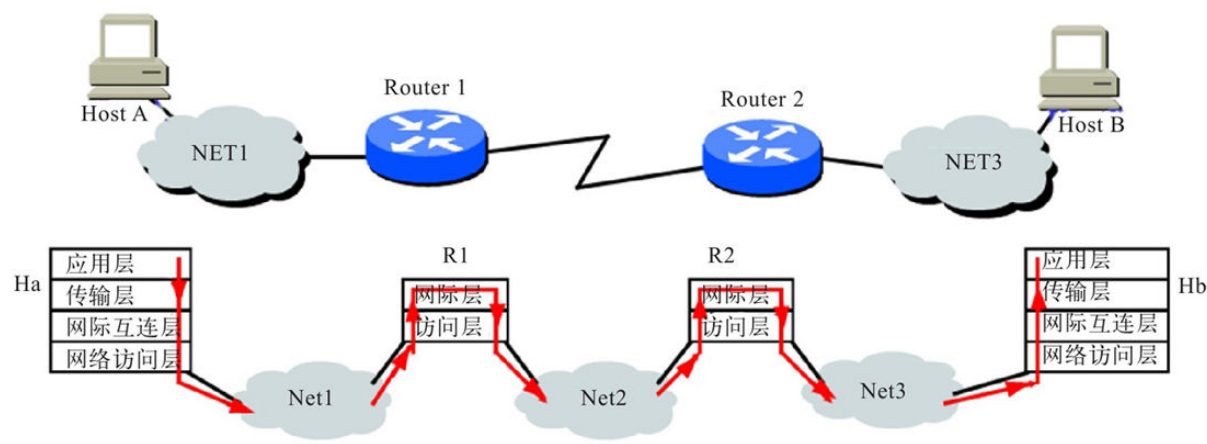


图 3-11 TCP/IP网络中的数据通信流程示例

总体来说，在各计算机网络体系结构的基本通信过程中，数据在发送端是自上而下传输的，而在接收端是自下而上传输的，整个通信过程必须依次经过通信发起层及以下各层，且不能跨越。但是各层只能与对方的对应层进行通信，不能出现错位。

3.3 网络体系结构的设计考虑

无论哪种网络体系结构，在设计时都不是随意的，而是经过无数专家学者充分认证和谨慎考虑得出的。我们只有在理解了设计者的设计思想后，才能更好地理解并在实际的网络系统设计、网络故障排除中应用这些设计思想。

3.3.1 网络体系结构中的层次划分依据

“网络体系结构”是一种概念上的蓝图，描述了整个网络的层次结构和基本的数据通信规则。实际上就是描述了整个网络中两个节点间实现有效通信的所有过程，然后将这些过程划分为逻辑上的组，而这些组就是网络体系结构中所说的“层”。

1.网络体系结构的设计考虑

网络体系结构设计时的考虑其实与一个公司在设计组织架构时要考虑整个公司的生产、经营流程应该怎样更有效是一样的（图3-12所示的是一家小型生产企业的典型组织架构）。在为新公司设计组织架构时首先会考虑这家公司的经营性质，在经营过程中要完成哪些任务，而这些任务应该由谁来完成，按什么顺序完成。这里的“谁”不是指具体的人，而是后面要设计的各个职能部门。这里的部门就相当于我们

所讲的网络体系结构中的“层”。在公司中，为了保证工作的顺利进行，每个部门内部的员工自然是紧密配合相互协调的，但任何一个部门都不可完全独立地工作，因为它们只是整个公司组织的一部分，必须要与其他部门协同工作。所以各部门都要有负责与上、下级部门之间沟通的负责人，这就相当于网络体系结构中的层与层之间相互通信的逻辑接口，称之为SAP（访问服务点）。

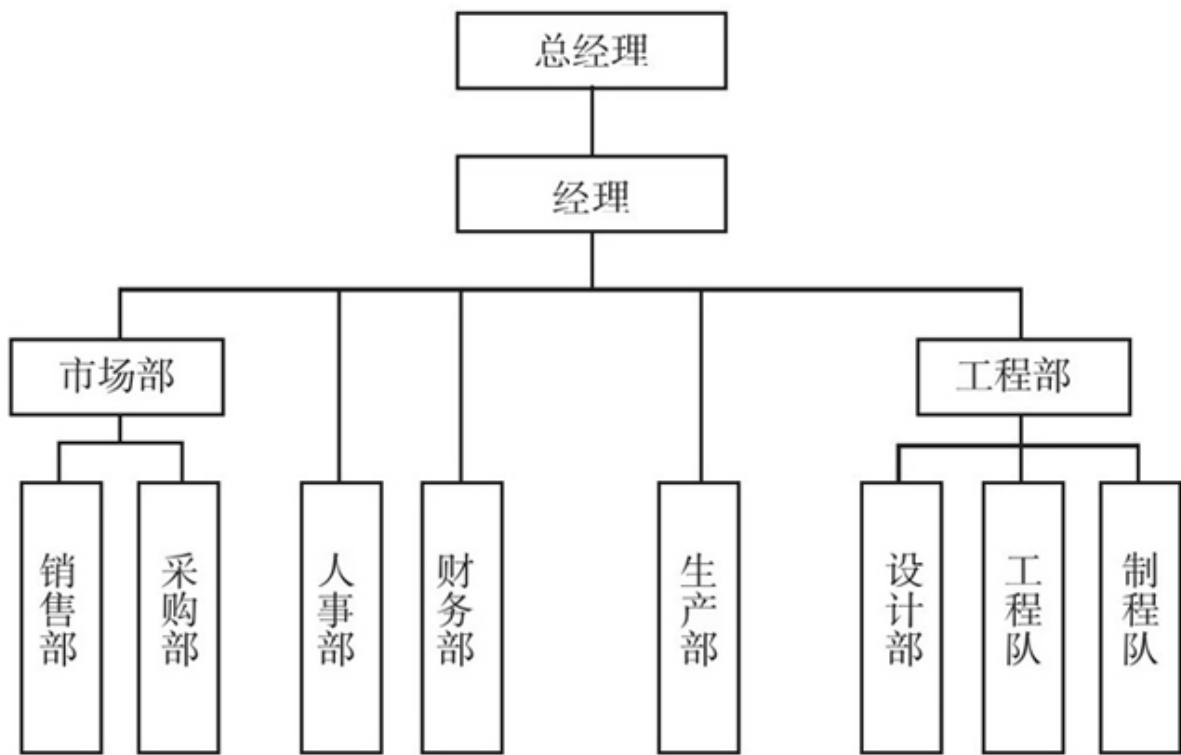


图 3-12 典型生产型企业的组织架构示例

既然计算机网络体系结构是分层的，那么不同层次之间必定有一个高低之分。注意，这里所讲的层次高低并不是从管理角度来理解的，可以说是仅从通信流程，或者说是服务调用关系上来划分的，主

要是“谁先谁后”的问题。就像在设计公司组织架构时要考虑命令上传、下达执行部门的先后次序一样。在各种计算机网络体系结构中，不同层次之间并不存在管理与被管理，只有通信流程角度上的先后次序，当然下层是为上层服务的。这就与工厂生产流水线的各个工作岗位只有先后次序之分，没有管理与被管理之分的道理是一样的。

2.网络体系结构中的层次划分

为什么要划分层次?其实原因很简单的，就是想把一个难以实现的复杂问题分解成多个容易实现的小问题。这在我们学习时也一样，要真正把计算机网络专业学好不是件简单的事，要学的知识和技能太多，涉及面太广。如果没有一个全局的观念，不采取分模块学习方法，是很难学得系统、学得全面的。学习时，如果我们把要学的知识和技能分成一个个比较小的模块，然后采取“个个击破”的方式进行学习，就可以比较快地全面掌握计算机网络专业所需学习的知识和技能。

OSI/RM是第一个标准化的计算机网络体系结构，划分了七个层次；随后IEEE又在其颁布的IEEE 802.1标准中发布了专门针对局域网的体系结构，也有专门针对无线局域网（WLAN）而发布的IEEE 802.11标准又定义了WLAN的体系结构，它们都只划分了两个层次；而最开始应用于ARPANET，随后成为事实上的当前Internet体系结构标准的TCP/IP协议体系结构划分了四个层次。这些不同网络体系结构的具体

层次将在本章后面具体介绍，但由此仍然可以看出，不同网络体系结构所包含的层数不一样，这主要是因为在不同网络中实现网络通信所需要的功能不一样。但无论是哪种体系结构，都直接或间接地包括了最低的两个层次，那就是“物理层”和“数据链路层”，因为这是所有网络通信的基石和物理通道。这些体系结构中的具体层次将在本章后面具体介绍。

当然，在这里不得不强调的一点就是，这些网络体系结构中的层次划分不是随意的，而是经过科学家严格的认证，经过深思熟虑后得出的。那么这些网络结构中的层次是依据什么来划分的呢？主要还是针对不同网络环境中，用户双方进行网络通信的流程，或者通信原理来对不同网络功能进行分层的，就像把一个产品的生产流水线划分为多个工序一样。另外，体系结构中的层次划分还要考虑到通信效率和可行性等诸多因素，既要使得整个网络体系结构尽可能简单，又要确保各层的功能和不同层次之间的协商容易实现。如生产一个比较简单的产品（如组装一台小风扇），它既可以只设一个工序，由一人来完成所有的工作，又可以细化分成两个，甚至更多工序来协同完成。这就需要综合考虑生产效率和生产成本等多个方面因素了。太粗了会影响生产效率（毕竟一个人做的速度有限），太细了不仅生产成本会大大增加（人工和设备费用比较高），也会影响生产效率（浪费在中间等和产品生产流程上的时间比较多）。网络体系结构层次的划分也是一样。

局域网体系结构和WLAN体系结构都是专门针对局域网内部的网络通信来设计的。而用户的具体网络应用所需的网络层及以上各层功能，完全可以直接通过局域网内部的用户主机操作系统来实现，所以也就只有“物理层”和“数据链路层”这两层了。而OSI/RM和TCP/IP协议体系结构都是针对不同网络之间的互连和通信而开发的，中间会有许多网络层或以上各层设备，所以它们不可能只有物理层和数据链路层，还得考虑网络地址的寻址（也就是网络层），考虑在不同计算机网络中如何建立通信连接和数据传输通道（也就是传输层，以及OSI/RM中的会话层和表示层），以及基于网络地址的各种网络应用（应用层）。

3.主机中所包含的层次功能

说到这里，可能马上有读者会问，难道在局域网内部就没有网络应用，不需要进行数据传输，不能使用IP地址这类三层功能吗？当然不是的，因为进行网络应用和数据传输是任何计算机网络（当然也包括局域网）的最基本用途，否则计算机网络就没有价值了。其实这里我们要充分认识局域网中各计算机的角色了。

现在计算机中安装的操作系统都是网络操作系统，都具有网络功能，可以提供OSI/RM体系结构中的各层功能。尽管局域网中的主机都在同一网段中，无须“网络层”的路由支持，但是计算机中的操作系统仍然支持“网络层”中的IP地址标识。而且事实上，两个远程网络间的通

信，中间提供网络连接的设备（如路由器设备）都只支持OSI/RM体系结构的下三层，高四层功能基本都是由各计算机提供的，如图3-13所示。

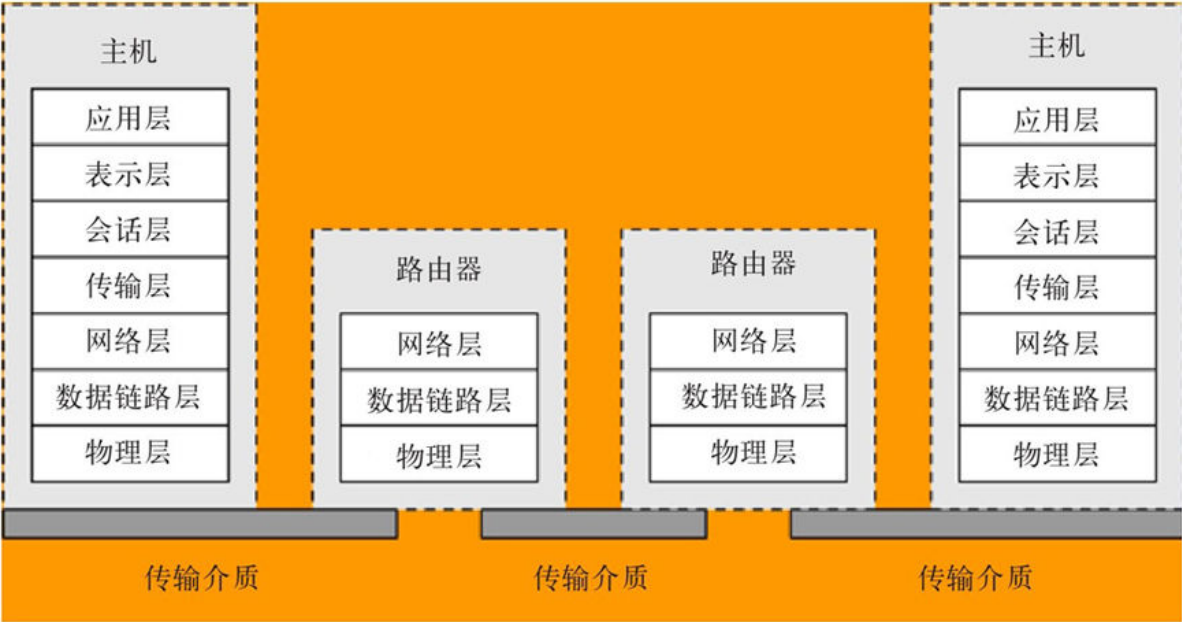


图 3-13 “主机层”所处的位置示例

3.3.2 网络体系结构分层的好处

我们已经知道，网络体系结构是分层的（但要注意，网络体系结构只是一个逻辑上划分的结构，真正起作用的还是各个网络实体，也就是网络设备软/硬件系统和用户进程等），但这样分层的体系结构到底有什么好处呢？综合起来表现在以下几个方面：

(1) 便于方案设计和维护

这是分层网络体系结构的最重要的优点。对于一个复杂的网络通信过程，如远程的计算机访问或网络访问，在通信过程中可能要经过许多中间结点，也可能要进行多种信息格式的转换，需要多种协议共同完成。如果没有一个分层结构，进行网络系统设计时就没了思路，到底该设计哪些功能，每个功能的实现又需要依赖哪些协议和服务，如何实现这些服务功能等都不是很明确。

有了分层结构后，在我们进行网络系统设计时，程序开发者就可以根据体系结构来理解计算机网络通信的流程，并看看在每一层中需要实现哪些类型的功能。这时，负责各层功能的程序开发者可以选择自己最擅长的语言或技术来进行开发，而不用管其他层的功能是如何开发、实现的。然后再通过一些服务接口就可以实现一个完整的网络通信任务了。这样一来，就可以把一个大的网络通信任务分解成几个

模块，然后采取个个击破的方法，很容易就可以实现这个复杂的网络通信任务了。

在维护方面也一样，如果没有分层结构，一旦出现网络故障，就可能要对整个网络的各方面进行故障分析和排除，就像大海捞针一样，毫无目的可言，显然工作量很大；而有了分层结构后，我们就可以根据不同层次的功能和实现原理，以及故障特点来专门针对某一个或少数几个层次进行对应的故障分析和排除，这样显然容易多了。

(2) 各层相互独立，技术升级和扩展灵活性好

在网络体系的分层结构中各层是相互独立的，也就是没有管理和被管理的区分，也互不影响，每一层都不需要知道它的上、下层是如何工作和进行功能实现的，仅需要上层知道该如何调用下层的服务，而下层又该如何为上层提供服务即可。这一方面可以使得整个复杂的设计任务变得比较简单，另一方面可以非常灵活地实现网络方案扩展或技术升级。

我们知道，网络技术的发展是非常快的，如果不采用分层结构，则可能任何一方面的技术更新，都可能引起全局的配置改动；而有了分层网络体系结构后，就仅需对相应层次和对应的服务接口进行技术更新即可，其他层次仍然可以保持不变，不受该层的影响。同样，还

可仅针对某一个或几个层次的服务进行修改，甚至在不需要时取消该层的服务。

（3）促进标准化

通过网络体系结构的标准化，可以统一各开发商的设计标准，实现协同开发，并允许不同厂家的产品想互通信，极大地促进了计算机网络的发展。

经验之谈 在进行网络体系结构分层时应使每一层的功能非常明确，而且比较容易实现。如果层次划分得太少，则会使每一层的协议太复杂，不利于实现；相反，如果层次划分得太多，每一层的功能又太单一，层与层之间的接口太多，同样会影响网络通信效率。这与我们在设计产品生产工艺时的考虑是一样的。通常每一层都要完成以下几方面的任务（可以是其中一种或几种）：

□差错控制：使得双方的网络通信更加稳定、可靠，不出现网络连接中断和数据丢失现象。

□流量控制：使得通信双方的数据发送和接收速率相当，以避免数据丢失。

□分段和重装：在发送端把大的数据块划分成更容易发送和接收的小数据块，然后在接收端按照对应数据块的发送顺序重新组装，恢

复为原来的大数据块。

□连接建立和释放：在进行正式的数据交换前必须先建立相应的链路连接，然后在数据交换完后立即释放所占用的链路，以提高链路的使用率。

3.4 网络体系结构中的通信协议

说到计算机网络体系结构，就不得不说计算机网络中的通信协议（也就是在标准中常说的“通信规程”）了，因为这些通信协议就是计算机网络体系结构中不同层次中的具体功能实现，也就是我们通常所说的各层所提供的“服务”。

3.4.1 理解计算机网络通信协议

通信协议对于计算机网络的重要性是不言而喻的，可以说没有通信协议，就没有计算机网络，因为它是计算机网络中的“软件”组成部分。当然，事实上任何一种计算机网络设备都不可能仅有硬件（俗称“裸机”），必须安装对应的软件系统，而这些软件系统中就包含了所支持的通信协议。每一种计算机网络都有一套协议在那里支持着。由于现在计算机网络种类很多，所以现有的网络通信协议的种类也很多，如常见的IEEE 802.3、PPP、PPTP、DNS、DHCP、HTTP等。那么什么是计算机网络通信协议呢？其实很简单，只要联系到我们平常与别人签的协议是什么就可以理解计算机网络中的通信协议了。

在我们平常工作和生活中与别人签订协议的目的就是约束双方或者多方的行为，达成对某一方面或者多个方面的共识。计算机网络通

信协议也是这样的，它就是约束通信双方在利用某协议进行通信时必须遵从对应的规则和约定，以达成通信共识，否则彼此各不相认，还如何通信？这就像一个只会中文，一个只会英文的两个人对话，肯定不能对话成功，因为他们相互都不明白对方在说什么。再比如，如果有几个人共同开发一个方案，结果他们各自使用的程序开发语言完全不同，且不能相互兼容，这样最终肯定是不能形成完整方案的。所以，计算机网络通信协议可以理解为，为了使网络中的不同设备能进行协同的数据通信而预先制定的一整套通信双方相互了解和共同遵守的格式和约定。

计算机网络的发展经历了漫长的探索过程，因此实际并不是先制定好了统一的标准再研究网络的。在网络发展中有很多国际组织和跨国公司都在致力于网络通信协议的制定，并产生了多种计算机网络体系结构模型和网络通信协议，而且有更多的组织和公司参与了开发和完善网络协议的工作。但要注意的是，通信协议并不一定是标准，也可能仅是某个公司为它们的网络设备专门开发的通信协议，不能用于其他类型的计算机网络中，也不能与其他厂家的设备互连。

如果你对网络设备有所了解的话，就一定清楚，Cisco（思科）有许多自己的协议，如CDP（思科发现协议）、DTP（动态中继协议）、VTP（VLAN中继协议）、IGRP（内部网关协议）和HSRP（主机备份路由器协议）等，这些协议是不能在其他厂家的设备上使用的

（除非得到思科的授权），也不能与其他厂家设备互连。只有通过国际上的一些标准化组织（如ISO、IEEE、IEC之类的）认可后才能以标准形式发布，如ISO/RM本身也是一个协议，但因为是由ISO制定并发布的，所以成为了国际标准，还有IEEE的一系列以太网、城域网、WLAN等协议也是公认的国际标准。

3.4.2 网络通信协议的三要素

网络体系结构中的各项功能服务，都是通过具体的通信协议来实现的。要实现有条不紊地进行数据通信，通信双方就必须事先约定好相应的规则，就像我们平常与人合作做一件事时事先谈好合作的方式和注意事项一样。当然，计算机网络中的数据通信规则不是通信双方具体签订的，而是由双方所采用的网络通信协议来完成的。

在利用网络通信协议进行计算机网络通信时主要涉及以下三个方面的问题：一是要实现什么样的网络服务，二是如何实现这些网络服务，三是如何与对方实现协同工作。这三个方面也就对应了计算机网络通信协议的三个基本要素：语义、语法和同步。下面分别解释它们各自的功能。

1.语义

“语义”可以理解为“语意”，是用来解决“做什么”的问题，也就是描述该通信协议具体用来完成什么功能。如我们平常工作和生活中所签的协议中的标题，或者主题一样，让人家一看就知道该协议是用来做什么的。双方在对等层次进行通信时，首先就要确定双方所使用的协议的“语义”是否一样，也就是完成的功能是否一样。如一方用户使用的是L2TP协议进行VPN通信（也就相当于他说“我要进行L2TP VPN

连接”），而另一方使用的却是PPTP协议来进行VPN（也就相当于他说“我要进行PPTP VPN连接”），目的都不一样，自然是不能协商成功的。道不同，不相为谋嘛。

在通信协议的“语义”部分还包括通信协议的版本，就像我们平常所签的协议的版本（主要是根据签协议的时间先后来判断）一样。不同的版本，所能实现的功能，以及实现的方式都可能存在大的差别，当然通常相邻版本的通信协议是向下兼容的。

2.语法

“语法”是用来规定通信时的信息格式，包括数据及控制信息的格式、编码及信号电平等，是用来解决“如何做”的问题的。如我们平时所签的协议中规定的具体条款，规定如何确保达到最终目标一样。如果双方进行网络通信时所用的通信协议一样，或者只要双方所使用的通信协议的“语义”部分是一样的，就相当于双方的目标是一致的。接下来的问题就是要如何来实现双方共同的目标，完成相应的网络服务。

通信协议也是软件，也是由一些计算机程序语言来开发的，所以协议的具体语法依据所采用的程序开发语言的语法规则来进行。尽管有时，双方所使用的通信协议名称并不一样，但只要其语义部分相同，且双方都能识别对方的语法规则，仍然可以进行通信。就相当于

我们与别人对话时，如果双方都懂几种语言，尽管母语不一样，但是都能够听懂对方在说什么，明白对方要做什么，仍然可以进行交流、共同完成某项任务。

3.同步

“同步”是用来解决“做的次序”问题的，也就是通信双方要完成某项网络服务，必须依据什么样的流程，匹配什么样的速率、什么样的电平来进行。在网络通信中同步又称为“握手”。通信协议是用来实现某项网络服务的，但在通信双方必须保持一定的程序执行步骤，就像打电话时的一问一答模式一样。否则，若一方的请求得不到对方的应答，就会出现程序错误，最终导致通信失败。

同时，因为程序执行的各个步骤之间是有相互依赖性的，有时要依据上一步骤的用户选择来做出下一步的选择。再有，还要考虑通信双方“说话”的速度问题。如因某些原因，一方的程序执行效率比较高（如网络带宽高，设备性能好），另一方的程序执行效率比较低（如网络带宽低，设备性能差），这时就得双方协商好，如何保持一个双方都可接受的执行速率，这样才能做到步调一致，以防出现差错。其实这就是我们在后面将要经常提到的“流量控制”、“拥塞控制”功能。如果不协商好传输速率的话，一方传输完数据好久了，对方还在接收，这时可能会因为传输延时等问题而出现数据丢失，从而导致对方接收到的数据不完整。就像我们平时打电话一样，如果双方都讲个不

停，那谁也听不清对方说什么，或者一方说话太快，对方根本听不清。

第4章 物理层

从本章开始，就要正式进入本书的主题了，这意味着将正式进入本书最难的部分。本书以图3-5所示的那个建议的五层体系结构为例，对网络体系结构各层进行全面、深入的分析。虽然这些技术看起来很深奥，但它们的确是计算机网络的基础知识，对于我们理解计算机网络通信原理，分析和排除计算机网络故障，以及设计网络系统等都是必要的。

通过对第3章的学习，我们已经知道，在所有计算机网络体系结构中的最低层都是“物理层”（尽管在TCP/IP协议体系结构中没有单独划分）。如果把整个计算机网络通信看成一个立体的层次模型的话，那么这个通信模型的底层就是“物理层”，因为计算机网络设备之间的连接必须依靠物理层的传输介质和相关协议进行。就像我们建造万丈高楼都离不开最底层的地基一样，不可能建造一个空中楼阁。同样，物理层也是所有计算机网络通信的必经之路，尽管这“路”有许多种，就像我们旅行时必须要有“路”（可以是泥巴路、柏油马路、水泥路、铁路等）可走一样，否则不要说到国外，就是出门都可能寸步难行。

本章围绕物理层的机械特性、电气特性、功能特性和通信规程这几个方面进行展开性的介绍，主要包括数据通信的基本模型、数据传输速率、数据传输类型、数据传输方式、数据传输模式、数据通信方

式、信号类型、信号编码、信号调制与解调、数据抽样、信道复用、传输介质和主要物理接口规程等。其中涉及许多非常复杂的技术原理，如信号编码、信号调制与解调、奈奎斯特准则、香农公式、数据抽样定律等，这些大家要着重理解。

4.1 物理层概述

计算机网络的“物理层”位于各计算机网络体系结构的底层（TCP/IP体系结构中的“物理层”功能是集中划分在最低的“网络访问层”中），负责在物理传输介质之上为“数据链路层”提供一个原始比特流（也就是数据是以一个个0或1的二进制代码形式表示的）的物理连接。但要特别注意的是，“物理层”并不是特指某种传输介质，而是指通过传输介质，以及相关的通信协议、标准建立起来的物理线路。也就是说“物理层”起到“修路”的作用，只不过这条路是用于计算机网络通信的“线路”罢了。

4.1.1 物理层的主要作用

讲到网络体系结构中的“物理层”，首先要知道的就是它到底有什么作用。相对其他各层来说，“物理层”尽管因传输介质类型、物理接口以及它们的通信规程（也就是“通信协议”）类型都非常多，所以包

括的技术和规程都比较多，但其主要功能相对来说还是比较单一的，具体包括：

（1）构建数据通路

“数据通路”就是完整的数据传输通道，可以是一段物理介质，也可以是由多段物理介质连接而成的。一次完整的数据传输，包括激活物理连接、传送数据、终止物理连接三个主要阶段。所谓“激活物理连接”，就是不管有多少段物理介质参与，在通信的两个数据终端设备间都要在电气上连接起来，形成一条可以在上面连续传输数据的通路。

（2）透明传输

物理层中可用的传输介质类型（如不同类型的同轴电缆、双绞线和光纤等）非常多，各自又有相应的通信协议和标准来支持，这就决定了不同的计算机网络可能有不同的“路”。物理层除了要把这些不同的“路”修好外，还要确保这些不同的“路”能“连通”起来，形成通路，最终实现把比特流传输到对端“物理层”，然后向“数据链路层”提交的目的。

要实现上述功能，需要物理层具有屏蔽不同传输介质类型和通信协议的功能，让进行网络通信的各方只看到有“路”可行，而不管修这些“路”所用的具体“材料”和相关标准，这就是物理层的“透明传输”功

能。有关这些传输介质类型和通信协议、标准将在本章后面具体介绍。

(3) 传输数据

无论是从网络体系结构中哪层发起的通信，最终的数据都得通过最低的“物理层”传输出去，因为这是网络通信的唯一物理通道。但“物理层”的传输单位是比特（bit，也就是“位”，数据中的一个二进制的0或1就代表1位）。“物理层”的基本作用是在发送端通过物理层接口和传输介质将数据按比特流的顺序传送到接收端的物理层。

(4) 数据编码

要使数据能在“物理层”上有效、可靠地传输，最关键的是要确保数据比特流能在对应的“信道”中正常通过。这就涉及“物理层”的数据编码功能，因为不同传输介质所支持的数据编码类型不一样（如归零码、非归零码、曼彻斯特码、差分曼彻斯特码等）。这些编码类型也将在本章后面具体介绍。

(5) 数据传输管理

“物理层”还具有一定的数据传输管理功能，如基于比特流的数据传输流量控制、差错控制、物理线路的激活和释放等。

4.1.2 物理层所定义的特性

总体来说，“物理层”的主要任务是定义与传输介质、连接器及其接口相关的机械特性、电气特性、功能特性和规程（也就是我们通常所说的“协议”）特性这四个方面。这些特性的规定都相当重要，因为有了统一的标准后，各设备生产厂家才能以相同的标准来设计、生产、制作这些传输介质、连接器、接口，并使用对应的通信协议实现彼此互连。

1.机械特性

物理层的“机械特性”定义了传输介质接线器、物理接口的形状和尺寸、引线数目和排列顺序，以及连接器与接口之间的固定和锁定装置。

在计算机网络中，我们常见的传输介质是同轴电缆、双绞线、光纤等，其实还有一种，那就是在路由器S口（串口）连接时，以及以前各种Modem与计算机连接时所使用的串行电缆。不管是哪种传输介质，在用于连网时都需要安装了连接器的网线电缆和设备或网卡中的对应接口。

细同轴电缆的连接器（称之为BNC头）有两种：一种是单一连接器，直接与BNC网卡或者与BNC头连接；一种是一种T形连接器，三端

分别用于连接细同轴电缆网线两端的连接器和BNC网卡，分别如图4-1a、b所示，制作成的细同轴电缆如图4-1c所示，BNC网卡如图4-1d所示。

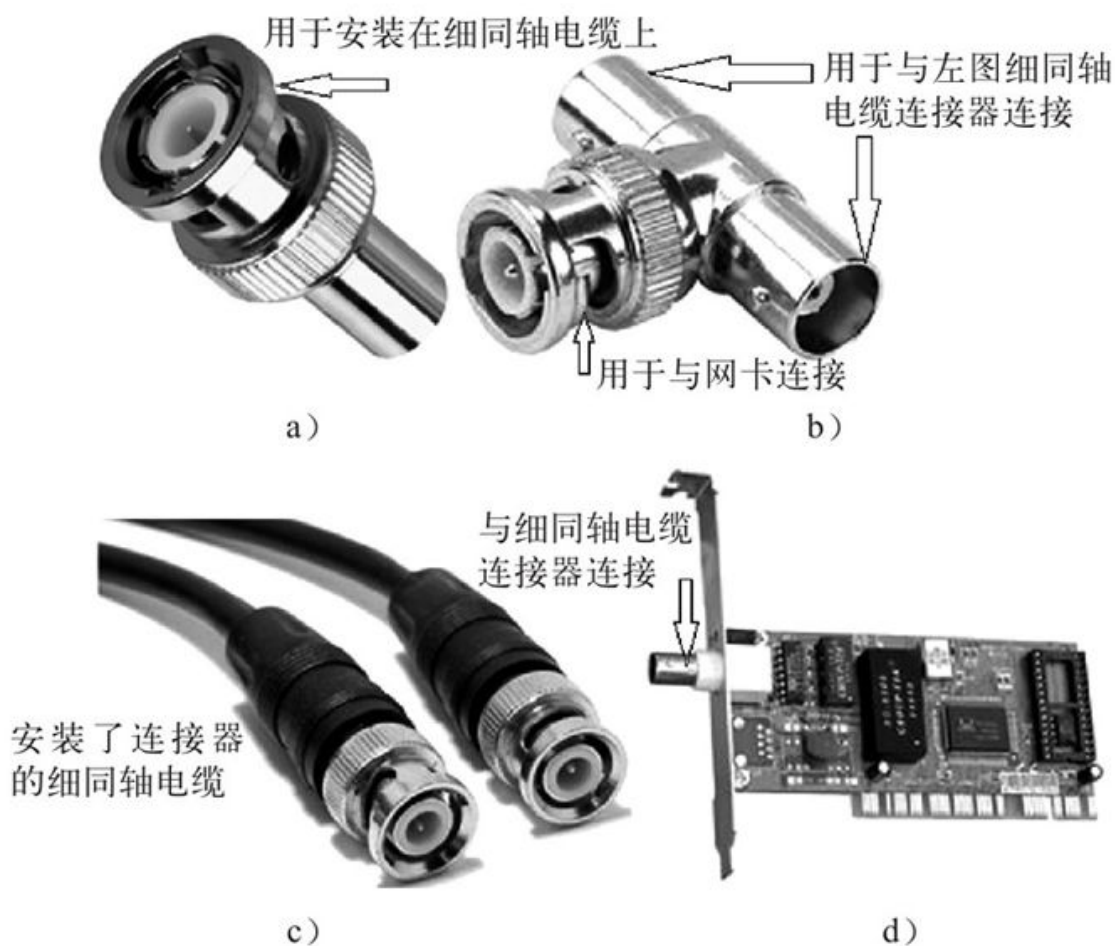


图 4-1 细同轴电缆BNC连接器、细同轴电缆网线和BNC网卡

对于双绞线以太网来说，连接器就是我们通常所说的“水晶头”（又称RJ-45接头），如图4-2a所示；安装了水晶头连接器的双绞网线如图4-2b所示，而网络接口（通常称为RJ-45以太网口）就是网卡或者交换机、路由器等设备的以太网端口，如图4-2c所示。

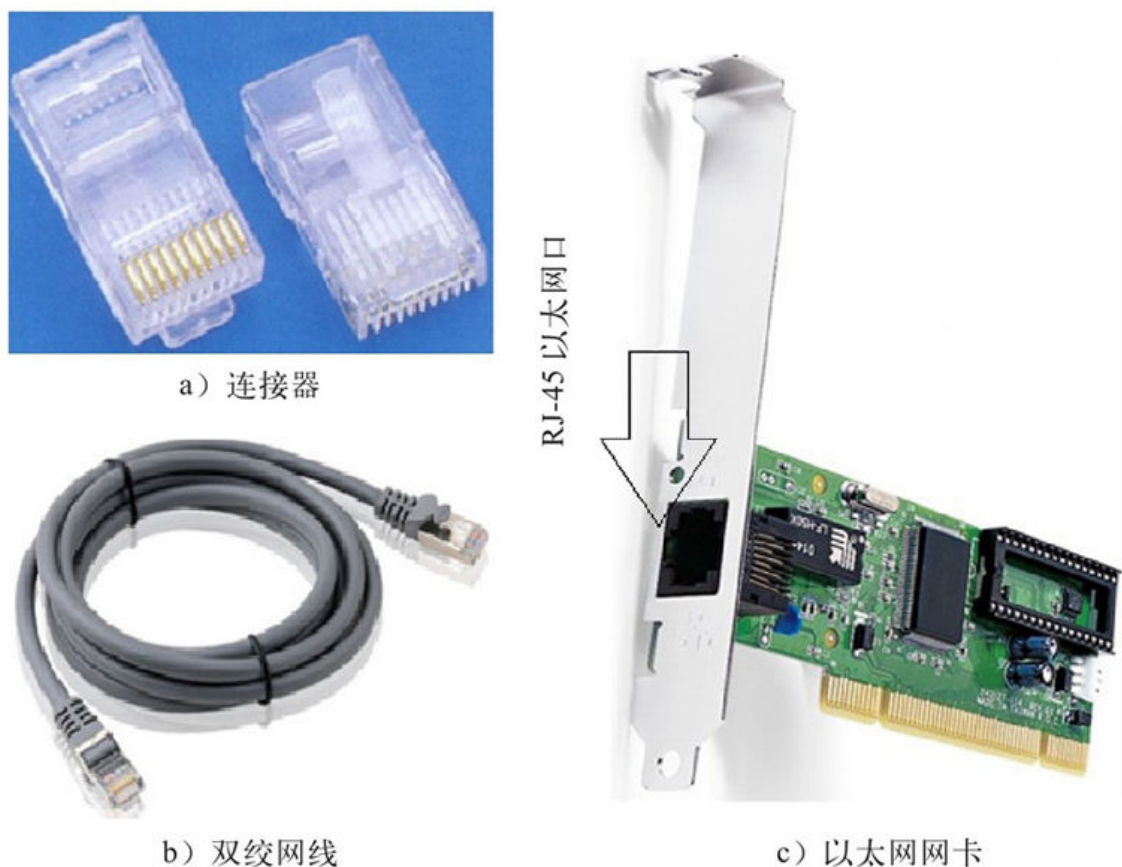


图 4-2 双绞网线连接器、双绞网线和双绞线以太网网卡

说明 因为光纤种类繁多，导致其连接器和接口种类也很多，在此就不举例说明了，本章后面介绍光纤时再做说明。

连接路由器的S口，和以前Modem连接PC时所用的串口统称为RS-232接口。它有多种类型的连接器（接口）；25芯连接器对应的标准为ISO-2110、EIARS-232C和EIARS-366A等，可用于串/并行音频调制解调器、公用数据网络接口、电报（包括用户电报）接口和自动呼叫设备中；34芯连接器对应的标准为ISO-2593，可用于CCITT V.25建议的

宽带调制解调器中；37芯和9芯连接器对应的标准为ISO-4902，可用于串行音频和宽带调制解调器中，与EIA RS-449标准兼容；15芯连接器对应的标准为ISO-4903，可用于CCITT X.20、X.21和X.22建议中规定的公用数据网接口中。

无论哪种类型的RS-232接口，都同时有用来连接DTE（Data Terminal Equipment，数据终端设备）设备的孔端连接器（也就是通常所说的“母头”）和用来连接DCE（Data Communications Equipment，数据通信设备）设备的针端连接器（也就是通常所说的“公头”）。图4-3所示的就是DB-25和DB-9两种连接器。制作好的电缆上都会一端是DTE连接器，另一端是DCE连接器。



图 4-3 DB-25和DB-9连接器

DTE是具有一定的数据处理能力和数据收发能力的设备，如PC机；DCE是在DTE和传输线路之间提供信号变换和编码功能，并负责建立、保持和释放链路连接的设备，如各种Modem。路由器一般同时

担当DTE和DCE双重角色，当然具体角色是由路由器上不同的接口担当的。DCE设备需要与DTE设备对接。

2.电气特性

物理层的“电气特性”规定了在物理连接上传输二进制比特流时线路上信号电压的高低、阻抗匹配情况，以及传输速率和传输距离限制等参数属性。早期的物理层电气特性标准定义的是物理连接边界点上的电气特性，而目前使用的新电气特性标准定义的都是接口发送器和接收器的电气特性，同时还给出了互连电缆的相关规程。目前所使用的新的物理层接口电气特性主要分为三类：非平衡型、差分接收器的非平衡型和平衡型。

这里所说的“平衡”与“非平衡”其实是根据发送器发送信号，或者接收器接收信号时信号电平是由单根线的电平值决定，还是由两根线之间的电平差值决定而判断的。“平衡”传输模式中都是采用一对线（除了地线之外）进行差分信号传输的，信号电平是由两根线上的电平差决定的，而“非平衡”传输模式则只用一根导线（除地线外）进行非差分信号的传输，信号电平仅由一根信号线上的电平决定。

说明“差分信号”是指两个物理信号间存在电平之差，又称“差模信号”，用一个差值表示。也就是指最终的信号电平不是由某一条导线上的信号电平来决定的，而是由两根导线的电平差值决定的。使用差分

信号的传输就叫做“差分传输”，区别于传统的一根信号线、一根地线的做法，差分传输是在两根线上同时传输信号，而且这两个信号的振幅相等、相位相反。在这两根线上的传输的信号就是“差分信号”。信号接收器通过比较这两个电压的差值来判断发送器发送的是逻辑0，还是逻辑1。但要注意，在电路板上，差分信号的两条导线的布线必须是等长、等宽、紧密靠近且在同一层面的两根线。只有这样才能做到尽可能使这两根导线上的差分信号同幅、反相、同步传输。

从严格意义上来讲，所有电压信号都是差分的，因为一个电压只能是相对于另一个电压而言的。在某些系统里，系统“地”被用作电压基准点。此时，这种信号规划被称为单端的，因为信号是用单条导线上的电压来表示的。一个差分信号作用在两条导线上，信号值是两条导线间的电压差，而且这两个电压的平均值还是会保持一致的，因为干扰会同时作用于两条导线上，并且效果一样。

我们用一个比喻来说明。“差分信号”就好比是跷跷板上的两个人，当一个人被跷上去的时候，另一个人被跷下来了，但是他们的平均位置（也就是那个支点）是不变的。这就是“差分信号”传输的原理和主要优势。因为这种反相、等幅差分信号传输方式可以有效地抑制外界干扰，具体原理将在后面的“平衡型”接口标准中介绍。

（1）非平衡型

“非平衡型”接口一般是指早期采用分立元件技术设计的接口，其信号发送器和接收器均采用非平衡方式工作。即每路信号仅使用一根导线传输（是“单线传输方式”），然后所有信号共用一根信号地线（发送器和接收器的地线是相连的，形成统一的接地），如图4-4a所示。



a) 非平衡型 (EIA RS-232C)



b) 差分接收器的非平衡型 (EIA RS-423A)



c) 平衡型 (EIA RS-422A)

图 4-4 三种物理层接口电气特性

这种非平衡接口中采用的是非差分传输方式，一路信号只在一根导线上传输，也就没有另一条导线上的等幅、反相的信号，所以这路

信号传输到了接收器后，它的电压也只由这条导线上的信号电平决定。这样一来，如果外界有一个信号进行干扰，就会直接叠加在这条导线的信号上，导致最终的信号失真。正因如此，在这种平衡型接口标准（典型代表为CCITT V.28建议的EIA RS-232C）中所采用的信号电平比较高，用+5V~+15V表示二进制“0”，用-5V~-15V表示二进制“1”。其目的就是为了使干扰信号对真正的信号产生较小影响，毕竟干扰信号电平通常不会很高。如果信号电平比较低的话，干扰信号的影响就会很严重了。

另外，同样由于使用的是单线传输，没有一个用来抵消各种干扰和串扰的线对，不同导线上的信号串扰也比较严重，所以在这种接口中，信号传输速率比较低（速率越高，线间的串扰越大。就像我们跑步一样，跑得越快，所产生的风对其他相邻跑道上的运动员干扰越大），规定在20kbps以内，连接两个接口的电缆长度限于15m以内（也是为了尽量降低线间串扰对有用信号的影响）。

（2）差分接收器的非平衡型

差分接收器的非平衡型（又称“新的非平衡型”）是原来“非平衡型”的改进版本，接口中的发送器采用的是上面介绍的非平衡工作方式（也就是一路信号也只用一条导线进行传输），但接收器采用“差分”工作方式，且发送器和接收器不是共用一条地线。在这种接口中，发送器仍使用一根导线发送信号，但在接收器端是两条导线输入的，

除了发送器直接驱动的那条信号线外，还有就是接自发送器地线的那条导线（如图4-4b所示），接收信号的最终电平是由这两条导线上的差值来决定的。

在这种新型的非平衡型接口标准（典型代表就是CCITT V.10/X.26建议的EIA RS-423A）中，发送器采用单线驱动，接收器采用双端输入，接收器最终信号的电平就是这两条线的差值（进行减法运算），所以称为“差分接收器”。如果没有干扰和串扰，接收器的最终信号就是发送器发送过来的那个信号电平；如果有干扰或串扰，那么这些信号会以相同振幅、相反相位叠加在这两条输入导线上，最终接收器上的差分结果就可以抵消这部分干扰或串扰信号。如现在发送器端发送一个5V的信号到接收器端，而同时在传输过程中有一个1V的干扰信号，那么在发送有用信号的那条导线上的电平就是 $5V+1V=6V$ ，而接到发送器地线的那条导线中，电平就会变为1V，最终在接收器端经过减法运算（因为要求它们的电平差），所得信号电平就为 $6V-1V=5V$ ，仍保持不变。

因为在接收器端采用了差分工作方式，可抵消外界干扰或线间串扰，所以它的信号电平可以比较低（这对电路中的芯片都是非常有利的），用+4V~+6V表示二进制“0”，用-4V~-6V表示二进制“1”。同样传输速率也可以比较高，在10m以内的近距离情况下，传输速率可达300kbps，但当传输距离达到1000m时，信号传输速率只有3kbps以下。

(3) 平衡型

“平衡型”接口是目前最广泛采用的一种物理层接口，一般是采用集成电路技术设计的。它规定“发送器”采用双线平衡发送方式，而“接收器”则采用差分处理方式，也不共用地线，如图4-4c所示。

在平衡型接口标准（典型代表为CCITT V.11/X.27建议的EIA RS-422A）中规定，两条导线承载同样的传输，但它们是等幅、反相的，反相信号看上去就像源信号在镜子中的影像。接收端并不侦听任何一个实际的信号，而是检测两个信号的差值。当两个反相信号抵达接收端时需进行减法运算，因为互为反相，所以两者相减的结果就是第一个信号值的两倍。比如，在某个特定时间点，第一个信号为+5V，那么第二个信号就是-5V，两者相减 $5V - (-5V) = 10V$ ，相当于放大了原始信号。同样，由于接收器采用了差分运算，所以就像前面介绍的“非平衡型”接口中所诠释的差分运算原理一样，它也可以抵消外界信号干扰和线间信号串扰。因为如果有噪声加入到传输过程中，那么噪声将以同样的方式影响两个信号。

比如，在某个特定时间点，第一条导线上的信号为+5V，第二条导线上的信号就是-5V，假设噪声电平为+1V，则第一条导线上的信号受干扰后为+6V，第二条导线上的信号受干扰后为-4V。如果噪声为-1V，则第一条导线上的信号受干扰后为+4V，第二条导线上的信号受干扰后为-6V。这时候接收端取两个信号的差值。但在这里要特别注

意的是，不论噪声是正还是负，不论噪声有多大，在相减的过程中都被抵消了，出来的差值都是10V。再经过10V到5V的转换运算，接收端可以收到和发送端一模一样的信号。这种抵消的能力使得平衡型传输获得了超级的传输速率，同时具有较高的抗干扰能力。所以，在平衡型接口标准中也可以用较低的信号电平，规定用+4V~+6V表示二进制“0”，用-4V~-6V表示二进制“1”。这种接口的信号传输速率也可以比较高，当在10m以内的近距离传输时，速率可达10Mbps，而当传输距离达到1000m时，信号传输率在100kbps以下。

3.功能特性

物理层的“功能特性”是指明传输介质中各条线上所出现的某一电平的含义，以及物理接口各条信号线的用途，包括：接口信号线的功能规定，接口信号线的功能分类。

CCITT V.24建议采用一根接口信号线定义一个功能的方法，这一方法是规定接口信号线功能的主要标准之一。而CCITT X.24则建议采用每根接口信号线定义多个功能的方法，这种多重复用一根接口信号线的方法可以减少接口信号线的数量。接口信号线按功能一般可分为数据信号线、控制信号线、定时信号线和接地线等四类。信号线的名称可以采用数字、字母组合或英文缩写三种方式来命名。CCITT V.24建议采用数字命名法。

按CCITT V.24建议的接口信号线命名方法，DTE-DCE接口信号线的名称的第一位均为“1”，所以也有人将其称为100系列接口信号线。相应的，CCITT V.24建议用于DTE-ACE（Automatic Calling Equipment，自动呼叫设备）接口信号线的名称的第一位均为“0”，故又有将这种接口标准称为200系列接口信号线的说法。还有其他一些常用的接口标准，如X.25（分组型公用数据网DTE-DCE接口标准）、X.20（公用数据网起止式传输业务的DTE-DCE接口标准）、X.20 bis（连接在公用数据网上的V系列建议起止式传输DTE-DCE兼容性接口标准）等。

4.规程特性

物理层的“规程特性”指明利用接口传输比特流的全过程及各项用于传输的事件发生的合法顺序，包括事件的执行顺序和数据传输方式，即在物理连接建立、维持和交换信息时，DTE/DCE双方在各自电路上的动作序列。

4.2 数据通信基础

物理层是各种计算机网络体系结构的最底层，数据在网络线路上的实际传输都是在物理层线路上进行的，所以数据通信的本质其实就是物理层的比特流传输。在正式介绍物理层中的一些具体技术前，本节先来介绍在计算机网络中进行数据通信的基础知识，包括数据通信基本模型、数据通信的基本概念、数据传输类型、数据传输方式、数据传输模式、数据通信方向等，在下节将介绍一些物理层中的主要技术，如信道复用方式、数字编码方式、数字调制技术等。

4.2.1 通信子网与资源子网

我们在第2章学习计算机网络发展历史时已了解到，在第二代计算机网络时期的APPANET网络中就已把计算机网络划分成两层结构，那就是通信子网和资源子网，如图4-5所示。在后面虽然有了各种更加详细的分层结构，但是就整个计算机网络而言，仍可以从大的方向上划分这两层。

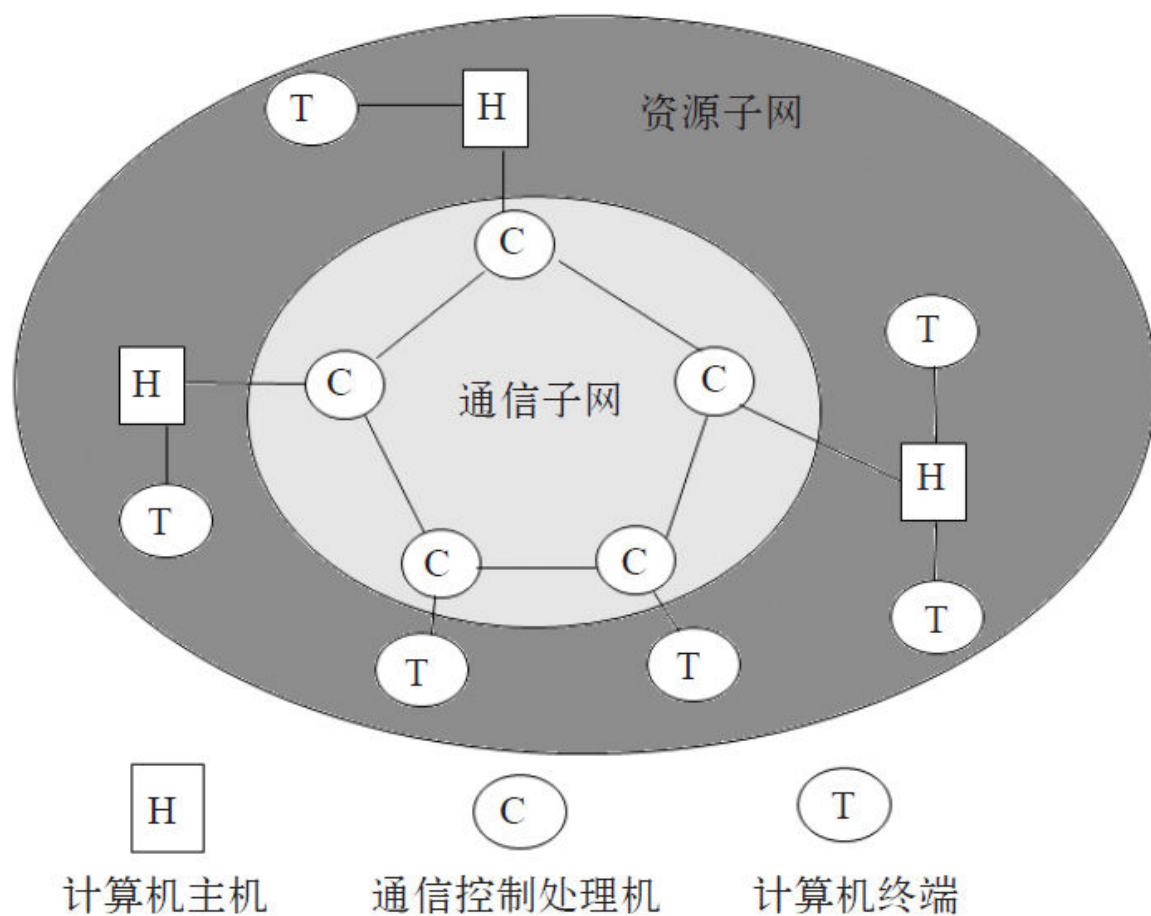


图 4-5 通信子网与资源子网示例

资源子网是由计算机系统、终端系统、连网的外围设备、各种软件资源与信息资源组成。资源子网主要负责全网的数据处理业务，向网络用户提供各种网络资源和网络服务。对于局域网来说，资源子网由连网的服务器、工作站、共享的打印机和其他外围设备、相关软件系统组成；对于广域网而言，资源子网由网上的所有主机及其外围设备组成。资源子网拥有所有的共享资源及所有的数据。

通信子网负责网络通信线路建立和通信处理，为网络用户提供数据传输、转发、加工和转换等通信处理工作，是整个网络数据通信的基础结构。对于局域网来说，通信子网由网卡、线缆、集线器、中继器、网桥、路由器、交换机等设备和相关软件组成；而对于广域网来说，通信子网由一些专用的通信处理机（如结点交换机、路由器）及其运行的软件和连接这些节点的通信链路组成（它们共同组成一个完整的通信线路）。

通信子网又可分为点-点通信线路通信子网与广播信道通信子网两类。广域网主要采用点到点通信线路，局域网与城域网一般采用广播信道。由于技术上存在较大的差异，因此在物理层和数据链路层协议上出现了两个分支：一类是基于点-点通信线路的，另一类是基于广播信道的。基于点-点通信线路的广域物理层和数据链路层技术与协议的研究开展得比较早，形成了自己的体系、协议与标准。而基于广播信道的局域网、城域网的物理层和数据链路层协议的研究开展得相对比较晚。

4.2.2 数据通信系统基本模型

数据通信是计算机与计算机或计算机与终端之间的通信。它传送数据的目的不仅是为了交换数据，更主要是为了利用计算机来处理数据。可以说它将快速传输数据的通信技术和数据处理、加工及存储的计算机技术结合了起来，从而给用户提供及时准确的数据。

在计算机网络数据通信模型中，涉及两个实体（源系统、目的系统）和一个通信信道（传输系统）。其中源系统又包括源站点、发送器，目的系统又包括目的站点和接收器。图4-6显示了计算机网络数据通信系统基本模型组成，同时以一个最简单、最典型的Modem通过PSTN（公共交换电话网络）拨号远程访问计算机为例，解释了这几个组成部分的基本作用。

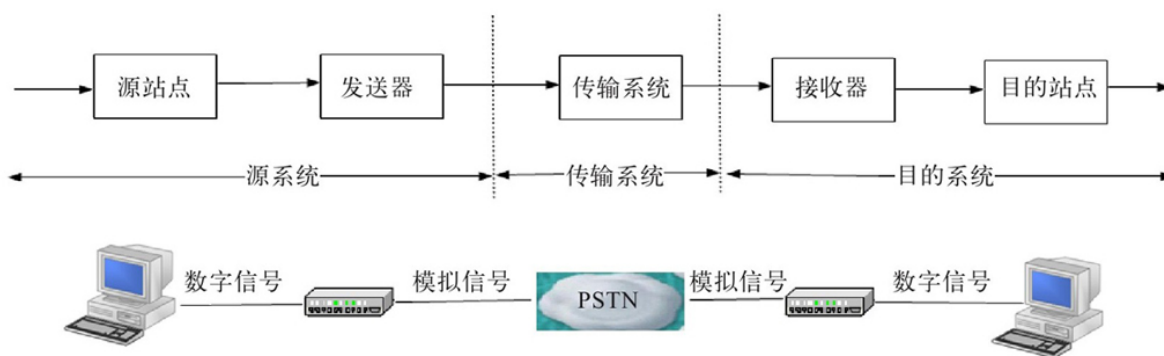


图 4-6 数据通信系统基本模型

1.源系统

“源系统”就是发送信号的一端，即发送方。它包括以下两个必需部分：

□源站点：产生要传输的数据的计算机或服务器等终端设备。

□发送器：对要传送的数据进行编码或者调制的设备，如各种调制解调器、计算机网卡。

2.传输系统

“传输系统”是计算机网络上的数据传输通道，除了源系统和目的系统外，网络中的其他部分都属于传输系统，包括通过物理层传输介质和相关协议建立的通信链路，以及结点设备，如线路上的交换机、路由器等设备。这部分是整个计算机网络中最复杂的一部分，不仅网络结构千差万别，而且网络通信协议和网络功能也是各种各样。

3.目的系统

目的系统就是接收发送端所发送信号的一端，即接收方。与“源系统”一样，它也包括两个必需的部分：

□目的站点：从接收器获取从发送端发来的数据的计算机或服务器等。

□接收器：接收从发送端发来的信息，并把它们转换为能被目的站点设备识别和处理的信息。主要也是各种调制解调器和网卡。

4.2.3 数据通信的几个基本概念

在计算机网络的数据通信中，信息、数据、信号和信道是四个最基本的概念。正确理解这四个概念，对于理解数据通信原理是非常重要的和必要的。

1.信息 (Information)

通信的目的是交换“信息”。“信息”是计算机网络中进行交换的一切原始内容的统称，可以是一串串的数字，也可以是各种文字，还可以是多媒体的图形、图像和语音。一般是字母、数字、语音、图形或图像的组合。但是我们知道，在计算机网络中传输的只能是二进制的数据，所以为了传输这些信息，首先需要将各个字母、数字、语音、图形或图像用二进制代码来表示，这就是我们通常所说的“信息编码”。在信息编码标准中，ASCII (American Standard Code for Information Interchange，美国信息交换标准代码) 编码被国际标准化组织接受，成为国际标准ISO646，又称为国际5号码。它既可用做计算机内部编码，也可用做数据通信中的编码标准。

在ASCII码标准中规定，一个字节为8位二进制，一个ASCII码占一个字节的低7位，最高位为校验位（用于传输过程检验数据正确性）。因为ASCII码用低7位二进制数表示一个字符，故这样一个字节可表示2

的7次方，即128种状态（从00000000~01111111）。每种状态与一个ASCII码字符唯一对应，这样一来，一个字节可以表示128个字符，其中包括26个英文大写字符、26个英文小写字符、10个数字字符、33个标点符号和33个控制符。例如大写英文字母A的ASCII码是65，小写英文字母a则是97。为便于书写和记忆，有时也将ASCII写作十六进制形式（一个字节恰好可以表示两个十六进制），即将某字符的ASCII码二进制数形式转换成十六进制数的形式，再标以H表示这是一个十六进制的数。例如字母A的ASCII码为01000001，写成十六进制即41H；字母C的ASCII码为01000011，写成十六进制即43H。

以上128个字符的编码规则如图4-7所示。例如大写字母C的ASCII码，只需在图中对应于字符C的位置，找出其横坐标 $D_6 D_5 D_4$ （代表第7位到第5位的最高三位，注意最低位是 D_0 ）和纵坐标 $D_3 D_2 D_1 D_0$ ，依次按 $D_6 D_5 D_4 D_3 D_2 D_1 D_0$ 的顺序排列出来，再在最高位补0，即得到C的ASCII码01000011。

$D_3D_2D_1D_0 \backslash D_6D_5D_4$	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	\	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	\$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(8	H	X	h	x
1001	HT	EM)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[k	{
1100	FF	FS	,	<	L	\	l	
1101	CR	GS	-	=	M]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

图 4-7 ASCII码编码图

2.数据（Data）

“数据”其实是上面介绍的“信息”的具体表示形式（任何要交换的信息最终都要以一个个具体的数据来传输），是许多信息通过某种方式组成的集合体。

数据有模拟数据和数字数据之分。模拟数据是通过连续取值得到的数据，如我们打电话时的通话语音，还有以前经常看到的模拟电影、模拟电视所用的数据都是属于模拟数据。而“数字数据”是把模拟数据离散、量化为二进制方式时取得的数据，如我们存放在计算机中

的的文件、图片、图像和软件等，以及我们现在经常看到的数字语音、数字电影、数字电视等所用的数据都属于数字数据。在计算机磁盘中存放在的必须是数字数据，而模拟数据通常存放在磁带、胶盘中。

模拟数据是时间的函数，并占有一定的频率范围，即频带。模拟数据也可以用数字信号来表示，那就是模拟信号的数字编码。对于声音数据来说，完成模拟数据和数字信号转换功能的设施是编码解码器（CODEC）。编码解码器将直接表示声音数据的模拟信号编码转换成二进制流近似表示的数字信号，在线路另一端的编码解码器则将二进制流码恢复成原来的模拟数据。数字电话通信是它的一个应用模型。

数字数据可以转换成模拟信号，如Modem可以把数字数据调制成模拟信号；也可以把模拟信号解调成数字数据。用Modem拨号上网就是现实中这一互换过程的应用模型。

“模拟数据”的大小一般是以时长为依据的，而“数字数据”是以各种容量为单位的。数字数据的单位主要有如下几种：位（bit，比特，指一个二进制的0或1）、字节（B，1B=8bit），而且1KB=1024B；1MB=1024KB；1GB=1024MB。

3.信号（Signal）

“信号”是“数据”在传输过程中电信号或光信号的表示形式，因为“数据”有“模拟数据”和“数字数据”两种类型，所以信号也有“模拟信号”和“数字信号”两种。“模拟信号”（analog signal）是“模拟数据”的电平信号表示形式，其信号电平是一个随时间连续变化的（中间没有跳跃）函数曲线（每个周期的曲线是一样的），如图4-8所示。而“数字信号”（digital signal）是“数字数据”的信号电平表示形式（由一个个“码元”组成，一个“码元”就包括一个脉冲周期），用两种不同的电平表示二进制的0和1信号所对应的电压脉冲，是离散（非连续变化）的，如图4-9所示。

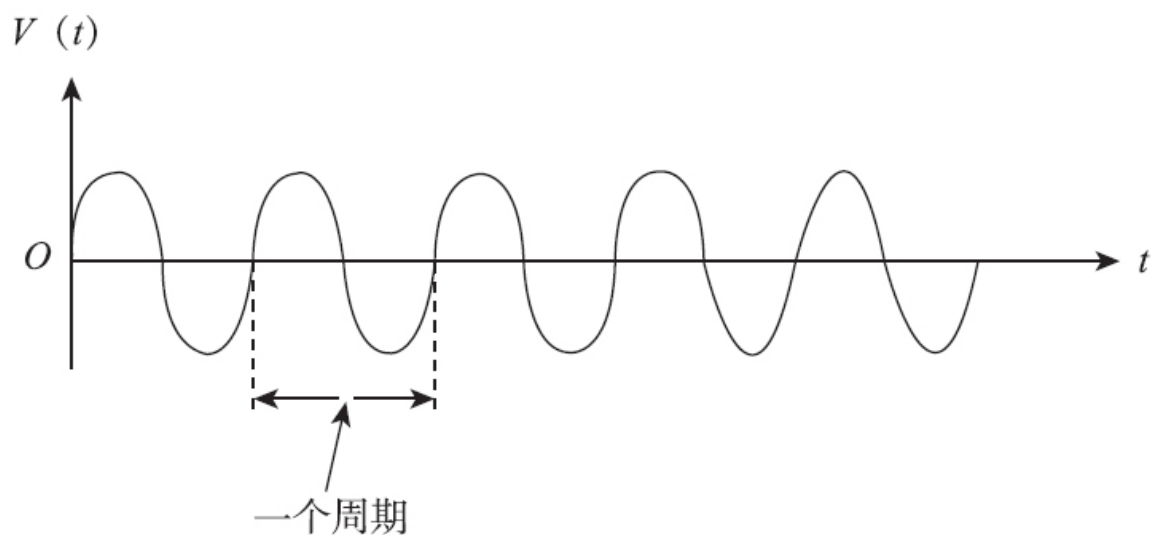


图 4-8 模拟信号波形

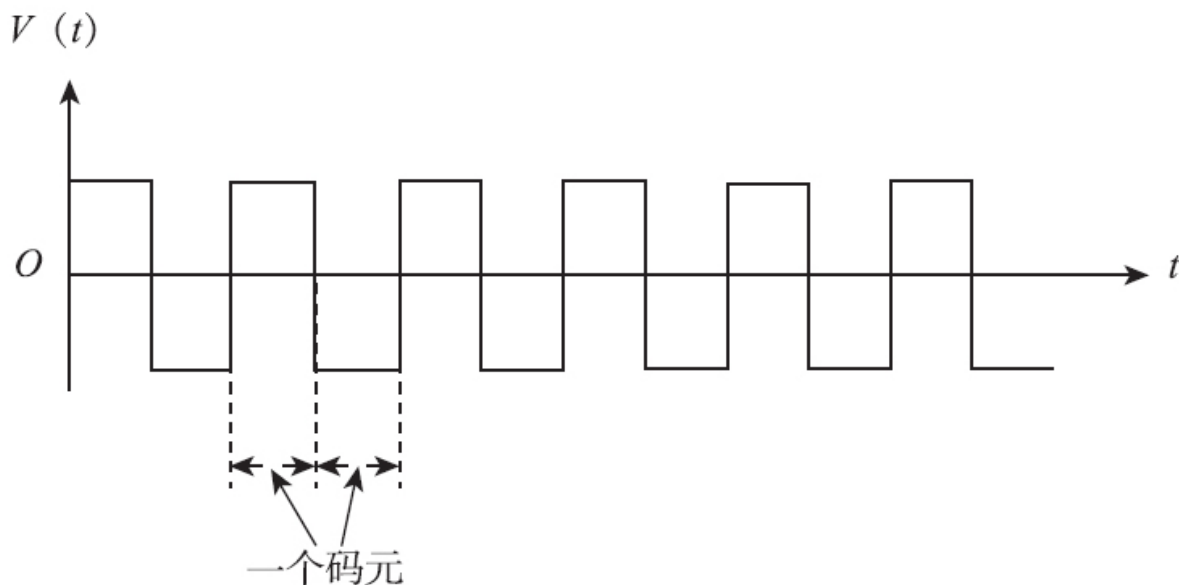


图 4-9 数字信号波形

4.信道

“信道”就是通信双方物理链路（包括有线物理介质上的链路和无线介质上的链路）上通过物理层协议建立起来的数据传输通道。默认情况下，一条传输介质就一条信道，也就是同一时刻只有一路通信。就像早期的公路，既不划分车道，也不划分通行方向，一条大路任由行人和车辆朝哪个方向通过。但目前广域网中，通常是采用“信道复用”技术实现在一条介质上划分多个信道，以实现同时多路、不同方向的通信（如图4-10所示），信道的具体复用原理将在本章后面介绍。



图 4-10 传输媒介中的信道

4.2.4 数据传输类型

数据通信的任务是准确迅速地传递信息。信源信号（也就是信号源发送的信号）经过信源编码之后成为离散的二进制数字信号。我们用一些离散的波形来代替这些数字信号。这些离散的信号可以直接进行传输，或者调制到载波上进行传输。这样就形成了两种最基本的数字信号的传输类型——基带传输和频带传输。在频带传输基础上，通过信道的复用又诞生了一种称为宽带传输的数据传输类型。

1.基带与频带

要理解什么是基带传输和频带传输，首先要理解什么是基带（base band）和频带（frequency band）。基带是指信源发出的，没有经过调制（进行频谱搬移和变换）的原始电信号所固有的频带（频率带宽），也称为基本频带。而频带是指对基带信号调制后所占用的频率带宽（一个信号所占有的从最高的频率到最低的频率之差）。有关调制技术和原理将在本章后面具体介绍。

2.基带信号和频带信号

信源发出的没有经过调制的原始电信号（原汁原味）就称为基带信号。基带信号的特点是频率较低，信号频谱是从零频率的直流成分

开始的。如果一个信号包含了频率达到无穷大的交流成分和可能的直流成分，则这个信号就是基带信号。

经过调制后的基带信号称为频带信号。频带信号的频率被限制在一个特定的频带中。如果一个信号只包含了一种频率的交流成分，或者有限几种频率的交流成分，我们就称这种信号为频带信号。根据原始电信号的特征，基带（或频带）信号可分为数字基带（或频带）信号和模拟基带（或频带）信号（相应的，信源也分为数字信源和模拟信源），具体由其信源类型确定。数字基带信号往往包含丰富的低频分量，甚至直流分量。在某些具有低通特性的有线信道中，特别是传输距离不太远的情况下，数字基带信号可以直接传输。

3.基带传输和频带传输

通过对以上基带信号和频带信号定义的了解，我们很容易知道，在信道中直接传输基带信号的方式称为基带传输，而在信道中传输频带信号的方式称为频带传输。

在基带传输中，由于不调制，所以整个信道只传输一种信号，通信信道利用率低。在基带传输中，需要在信源端对数字信号进行编码，在信宿端对编码的数据进行解码恢复，这分别是由编码器和解码器完成的。由于在近距离传输范围内基带信号的衰减不大，从而信号

内容不会发生变化。因此在传输距离较近时，计算机网络都采用基带传输类型。大多数的局域网使用基带传输，如以太网、令牌环网等。

而频带传输广泛应用于广域网中，因为在这种网络通信中，往往需要同时发送多种信号（如数据信号、路由信号，以及各种网络控制信号），这时就可以利用高频率的信号来调制低频率信号，以实现同步传输，也提高了信道利用率。由于频带传输中所用的传输频率通常比较高，所以也适用于远距离的广域网通信。有关信号调制将在本章后面具体介绍。

4.基带传输系统和频带数字传输系统

要进行基带传输，必须有一整套基带传输系统。图4-11所示的是数字基带传输系统的基本结构，主要由编码器、发送端低通滤波器、传输信道、接收端低通滤波器、采样判决器和解码器组成。此外为了保证系统可靠有序地工作，还应有位同步系统。

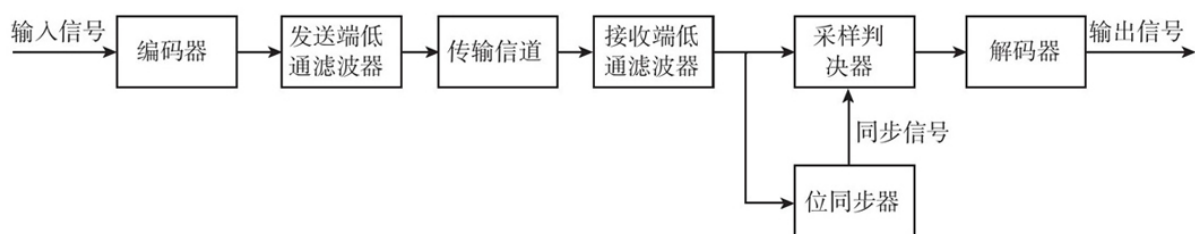


图 4-11 数字基带传输系统基本结构

图4-11中所示各组成部分的解释如下：

□编码器：将信源或信源编码输出的编码类型（通常为单极性不归零码NRZ）变为适合于信道传输的码型。有关基带信号的编码类型将在本章后面专门介绍。

□发送端低通滤波器：将编码之后的基带信号变换成适合于信道传输的基带信号，这种变换主要是通过波形变换来实现的，其目的是使信号波形与信道匹配，便于传输，减小码间串扰，利于同步提取和采样判决。

□传输信道：它是允许基带信号通过的传输介质，如双绞线、电缆线等。信道的传输特性通常不满足无失真传输条件，甚至是随机变化的，还会额外引入噪声。

□接收端低通滤波器：滤除带（基本频带信号）外噪声，并对信道特性均衡，使输出的基带波形无码间串扰，有利于下面的采样判决。

□采样判决器：在传输特性不理想或存在噪声的背景下，在规定的时刻（由位定时脉冲控制）对从接收端滤波器输出的波形进行采样，以恢复或再生基带信号。

□解码器：对采样判决器输出的信号进行解码，还原有用的输入信号，并使输出码型符合接收终端的要求。

□位同步器：提取位同步信号，通过同步信号控制采样判决器，再生数字基带信号。

同样，要进行频带传输，也必须有一整套频带传输系统。在前面介绍的数字基带传输系统中，为使数字基带信号能够在信道中传输，要求信道应具有低通（也就是所有低于某个频率的信号均可通过，高于某个频率的信号全部滤除）形式的传输特性。而大多数实际信道具有带通（也就是只有频率在某个范围内的信号才能通过，其他的均将滤除）传输特性，基带信号不能在这种带通传输特性的信道中传输，必须进行调制。调制就是把一个调制信号放在另一个频率更高的载波信号上，使在信道中传输的信号具有两种信号波的双重特性，然后通过调制信号的某些特性（如振幅、相位）影响载波信号相应特性的一种技术。

数字频带传输系统与数字基带传输系统的结构基本类似，不同的只是在频带传输系统的接收端要采用“调制器”替代基带传输系统中的“编码器”，采用“发送端带通滤波器”替代基带传输系统中的“发送端低通滤波器”，在频带传输系统的接收端要采用“解调器”替代基带传输系统中的“解码器”，采用“接收端带通滤波器”替代基带传输系统中的“接收端低通滤波器”，如图4-12所示。

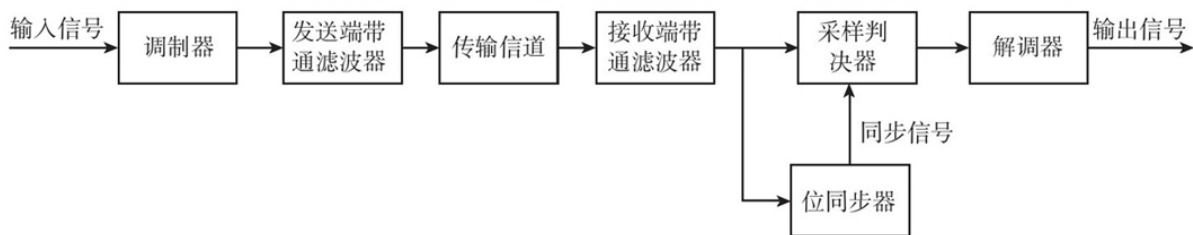


图 4-12 数字频带传输系统的基本结构

数字频带传输系统是发送端含有调制器，接收端含有解调器的数字通信系统。数字调制是用数字基带信号改变高频载波的参数，实现基带信号变换为频带信号的过程，此过程中信号频谱由原来的低频信号搬移到高频段。数字解调是把数字频带信号恢复成原来数字基带信号的过程，此信号中的频谱由高频段恢复到原来的基带信号的低频段。

由于基带传输的信号频率通常较低，为了能传输更远距离，抵抗外界干扰，在复杂的网络环境和远距离传输中，通常不采用基带信号，而是把发送端发出的原始信号经过调制，附加在一些高频率的载波上进行传输。这种频带传输不仅克服了目前许多长途通信线路中不能直接传输基带信号的缺点，而且能实现多路复用的目的，从而提高通信线路的利用率。

5. 宽带传输和宽带传输系统

所谓“宽带”（Broadband）就是采用比音频（4kHz）更宽的频带，该频带包括了大部分电磁波频谱，是相对前面的“频带”来说的，因

为“频带”只具有有限的频率宽度。使用这种宽频带进行的数据传输就称为宽带传输。宽带传输其实与频带传输具有相同的特点，它也需要对信源发出的原始信号进行调制，但它将一个信道分成多个子信道，分别传送音频、视频和数字信号，而且宽带传输中的所有子信道都可以同时发送信号。宽带传输一定是采用频带传输技术的，但频带传输不一定是宽带传输。

最常见的宽带传输就是我们天天用的有线电视，在一根同轴电缆中可以传输那么多电视频道。还有无线广播也是采用宽带传输的。宽带通常是传输模拟信号的，数据传输速率范围为0~400Mbps，而通常使用的传输速率是5~10Mbps。

各子信道的宽带传输系统结构与前面介绍的频带传输系统结构基本一样。

4.2.5 数据传输方式

计算机网络数据通信的数据传输方式可分“串行传输”和“并行传输”两种。

(1) 串行传输

“串行传输”指的是数据流以串行方式一位位地在一条信道上传输。如一个字符的8个二进制代码，需要由高位到低位顺序排列依次传输，如图4-13所示。等第一个字符的最高位传输完后，再传输第二个字符的最低位，依此类推，这样串接起来形成串行数据流。

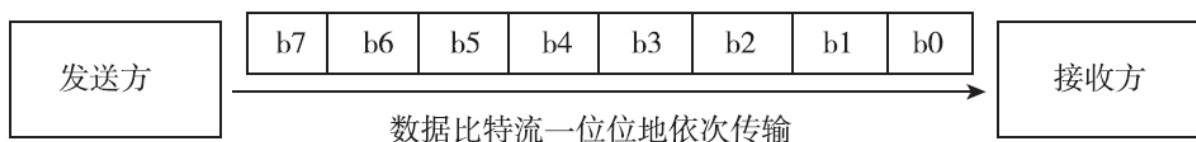


图 4-13 串行数据传输原理示例

很显然，串行传输只需要一条传输信道（当然在一个接口中可以有多条这样的串行传输信道），传输速度比较慢，但易于实现、费用低，是目前主要采用的一种传输方式。另外，串行传输存在一个收、发双方如何保持码组或者说字符同步（也就是如果识别一个字符包括哪几个比特位）的问题。这个问题不解决，接收方就不能从接收到的数据流中正确地区分出一个个字符来，因而传输将失去意义。如何解

决码组或字符同步的问题，就涉及本节后面将要介绍的异步传输方式和同步传输方式了。

采用串行传输方式的典型代表有计算机串行接口（简称串口，对应RS-232C标准），还有如USB接口、SATA（串行ATA）磁盘接口等，它们在传输数据时都是一位位地进行串行传输的。

（2）并行传输

“并行传输”指的是数据以一组或者整个字符的方式在多条并行信道上同时传输。常用的就是将构成一个字符代码的8位二进制码，分别在8个并行信道上进行传输，如图4-14所示。由于同一时刻多条信道上同时传送了整个字符的所有比特位，所以收、发双方不存在字符的同步问题，这是并行传输的一个主要优点。但是，并行传输必须有并行信道，这往往带来了设备上或实施条件上的限制，一般适用于计算机和其他高速数据系统的近距离传输。

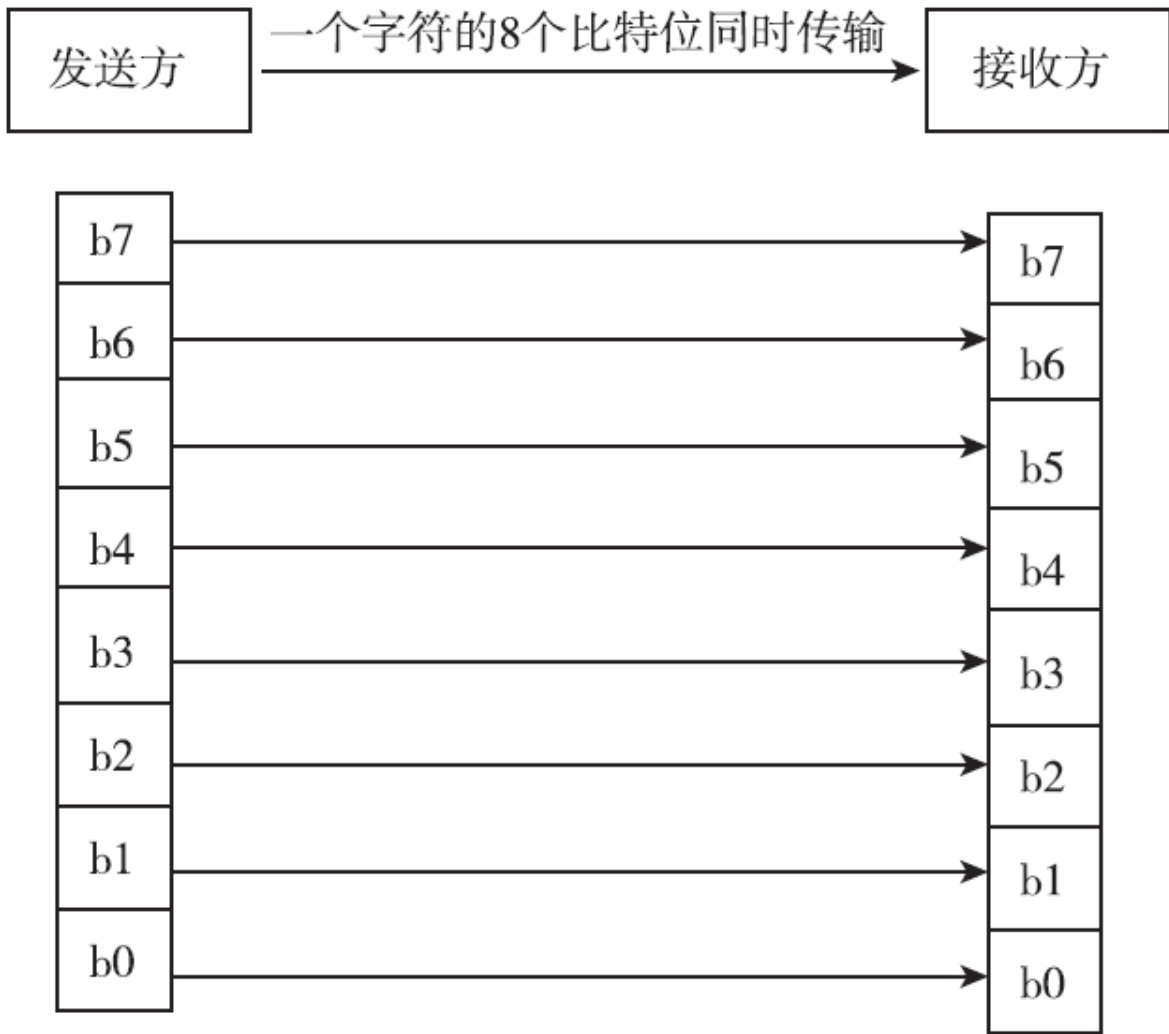


图 4-14 并行数据传输原理示例

采用并行传输方式的典型代表有计算机上连接打印机的并口、磁盘ATA（或者IDE）接口等，计算机主板上CPU与其他芯片的数据传输也是采用并行传输方式。

4.2.6 数据传输模式

在本节前面介绍串行传输方式时就已说到，串行传输接收端存在一个如何从串行数据比特流中正确地划分出发送的一个个字符的问题，也就是字符同步的问题。在串行数据通信中同步问题十分关键。发送端一位一位地把信息通过介质发往接收端，接收端必须识别信息的开始和结束，而且必须知道每一位的持续时间。只有这样，接收端才能从传输线路上正确地取出被传送的数据。根据实现字符同步的方式不同，有同步传输（**Synchronous Transmission**）和异步传输

（**Asynchronous Transmission**）两种方式。所以这里所讲的同步传输和异步传输都是针对串行传输模式而言的，并行传输模式中不涉及同步问题，因为并行传输中一次就可以传输整个字符。

1.同步传输

同步传输中关键是要理解“同步”的含义。同步就是指通信双方在传输过程中是同步进行的（也就是接收端与发送端同时开始工作，并且接收端按数据的发送顺序进行接收），同步的依据就是双方有相同的时钟参考，能同时开始数据的发送和接收。通常这个时钟参考是同步时钟线或同一个时钟源。

同步传输是一种以数据块为传输单位（通常是以“帧”为单位的），以相同的时钟参考进行数据传输的模式，因此又称为区块传输。在该传输模式下，每一比特数据的持续传输时间都是相等的，而且在每个字符的传输过程中，两个字符间传输所需等待的时间也是相同的。

另外，在同步传输中的数据块的开始和结尾部分都有一个用于数据帧同步的特殊字符、特定的字节或特定的帧，这具体要视所采用的同步方法而定。同步传输分为面向字符的同步传输、面向比特的同步传输和面向字节的同步传输等几种。面向字符的同步传输中，每个数据块的第一部分包含一组同步字符（一般是1~2字节，是一个独特的比特组合，相当于数据帧的“起始位”），用于通知接收方一个帧已经到达。它同时还能确保接收方的采样速度和比特的到达速度保持一致，使收、发双方进入同步。在传送完一个数据帧的最后一部分包含一个帧结束标记（称为区块结束字符，一般是1字节，也是一个独特的比特串，相当于数据帧的“停止位”），用于表示在下一帧开始之前没有别的即将到达的数据了。另外，一般还要附加一个校验序列（称为区块校验字符，一般也是1字节），以便对数据块进行差错控制。同步传输模式中，数据帧格式如图4-15所示。因为涉及“数据链路层”的帧同步，所以具体将在下章介绍。

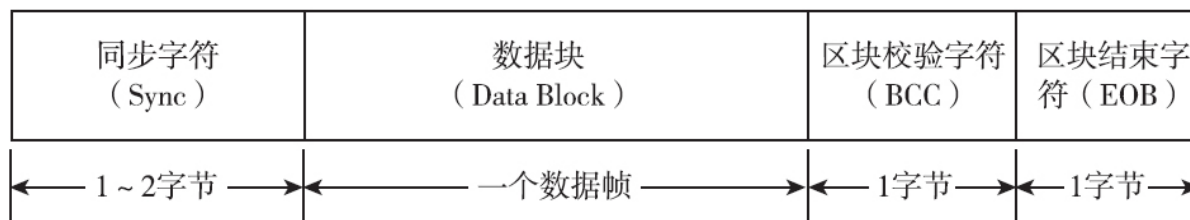


图 4-15 同步传输数据帧格式

采用同步传输的技术代表有SDH（Synchronous Digital Hierarchy，同步数字系列）、STM（Synchronous Transfer Module，同步传输模块）和HDLC（High level Data Link Control，高级数据链路控制）等。

与下面将要介绍的“异步传输”方式比较，“同步传输”在技术上较复杂，但不需要对每个字符加单独的“起始”和“停止”比特，接收方不必对每个字符进行开始和停止的操作，只需要在检测到帧同步字符时接收它们即可。而且同步传输是以帧为传输单位（异步传输是以字符为传输单位的，一个帧通常有500字节，4000比特）进行传输的，只需在一个数据帧的前后加上标志字符即可，因此传输效率高，常用于较高速的数据传输。

2.异步传输

异步传输的关键也是异步这两个字，是指通信双方没有相同的时钟参考（也就是发送端和接收端不同时开始工作），但双方在数据传输速率上是同步的。即指每个字符之间是异步的，但一个字符内的每一位还是同步的。

异步传输是以字符为单位进行数据传输的。在发送每一字符代码的前面均加上一个“起始位”信号（长度为1比特，极性为0），用以标记一个字符的开始；在一个字符代码的最后加上一个“停止位”信号（长度为1或2比特，极性为1），用以标记一个字符的结束。字符位占5~8位，具体取决于数据所采用的字符集，如电报码字符为5位，ASCII码字符为7位。异步传输时每个字符的数据结构如图4-16所示。图4-17所示是一个异步传输模式的数据传输示例。

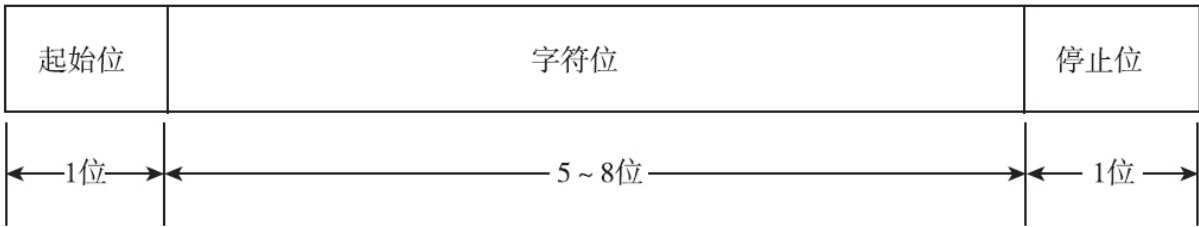


图 4-16 异步传输中的数据格式

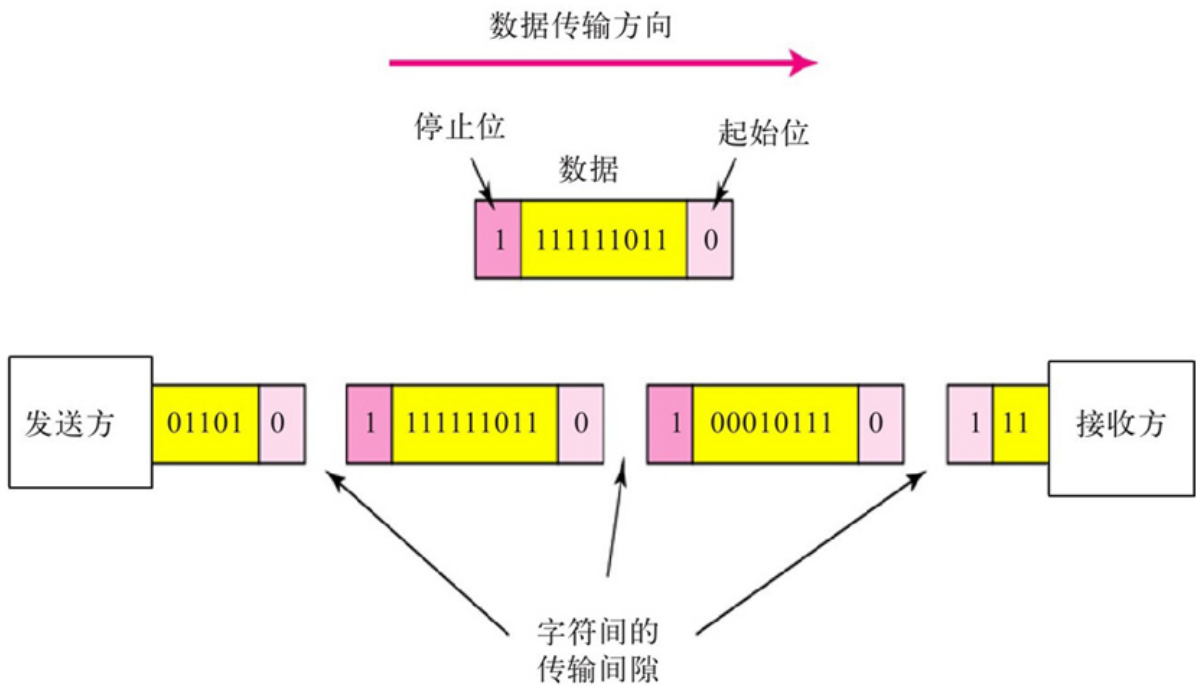


图 4-17 异步传输示例

异步传输的典型代表就是ATM（异步传输模式）技术，目前仍是城域网和广域网中主流的交换技术之一。早期还有一种B-ISDN（宽带ISDN）技术，也是异步传输技术的代表。

在“异步传输”中，发送方可以在任何时刻发送这些比特组，而接收方从不知道它们会在什么时候到达，故不需要同时开始接收。被发送的比特组需进行排除等候。最典型的一个异步传输实例就是我们访问互联网上的网站，点击一个网站，网站服务器一般不会马上响应，而是等它有资源响应时才响应，所以网站打开时往往有些延时。

异步传输的优点是字符同步实现简单，收发双方的时钟信号不需要严格同步。缺点是对每一字符都需加入“起、止”码元，使传输效率降低。例如假设每次传送7比特的信息，每个字符需要另外加一个起始位和一个停止位。另外，异步通信主要处理ASCII编码的信息，这意味着增加了第3个控制位，即奇偶校验位，这样每个字符附加3位控制位，传输效率只有70%。

4.2.7 数据通信方式

在数据通信中，按照通信双方的通信方式通信可以分为以下三种：单工通信、半双工通信、全双工通信。

1.单工通信

图4-18a所示为单工通信方式。在单工通信中，信号只能向一个方向传输，任何时候都不能改变信号的传输方向。

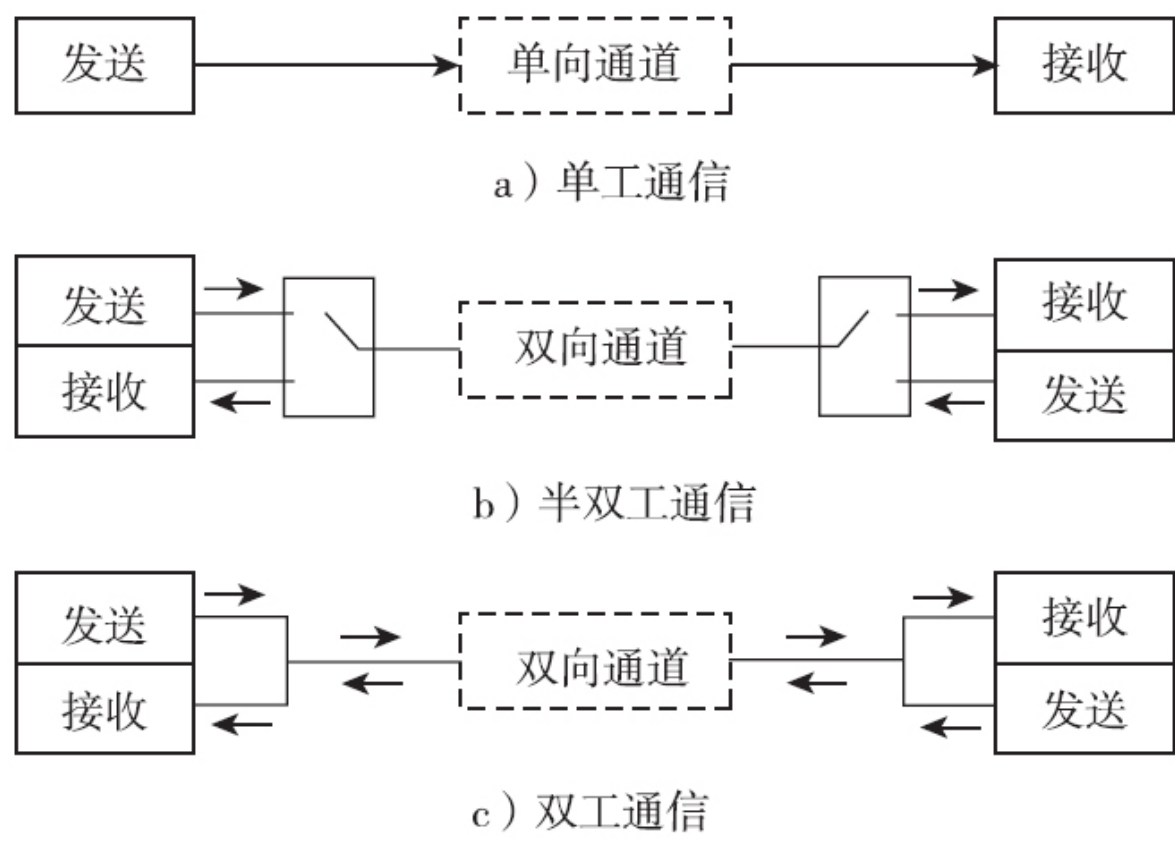


图 4-18 三种数据传输方向

“单工”（也就是“单向工作”）就是指永远只能往一个方向传输数据。数据发送方和接收方都是固定的。就相当于我们现实生活中所见的单向公路（通常是单车道）。单工数据传输就相当于汽车在单向车道行驶，如图4-19所示。在这样的公路上，车只能按一个方向行驶。很显然，这种数据传输方式效率最低，不能满足我们的数据通信需求，所以现在一般不用了。

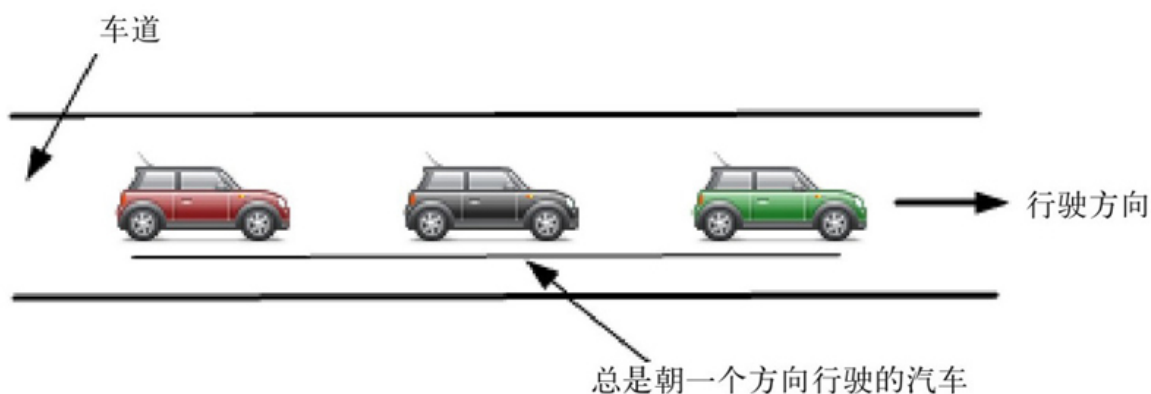


图 4-19 单工通信类比示例

2.半双工通信

图4-18b所示为半双工通信方式。此时信号可以双向传送，但必须是交替进行，同一时间只能向一个方向传送。也就是说，“半双工”在同一时刻只能进行单向数据传输，但是在不同时刻可以进行另一个方向的数据传输。这就相当于双向单车道。在这样的车道中，同一时刻只允许一个方向的汽车通过，只有当另一方向没有车过来的时候，才能从自己一方发车，如图4-20所示。它与前面的“单工”不一样的就是它

可以进行双向数据通信，只不过两个方向不能同时进行数据传输。半双工通信是目前网络设备所支持的最主要的数据传输模式。

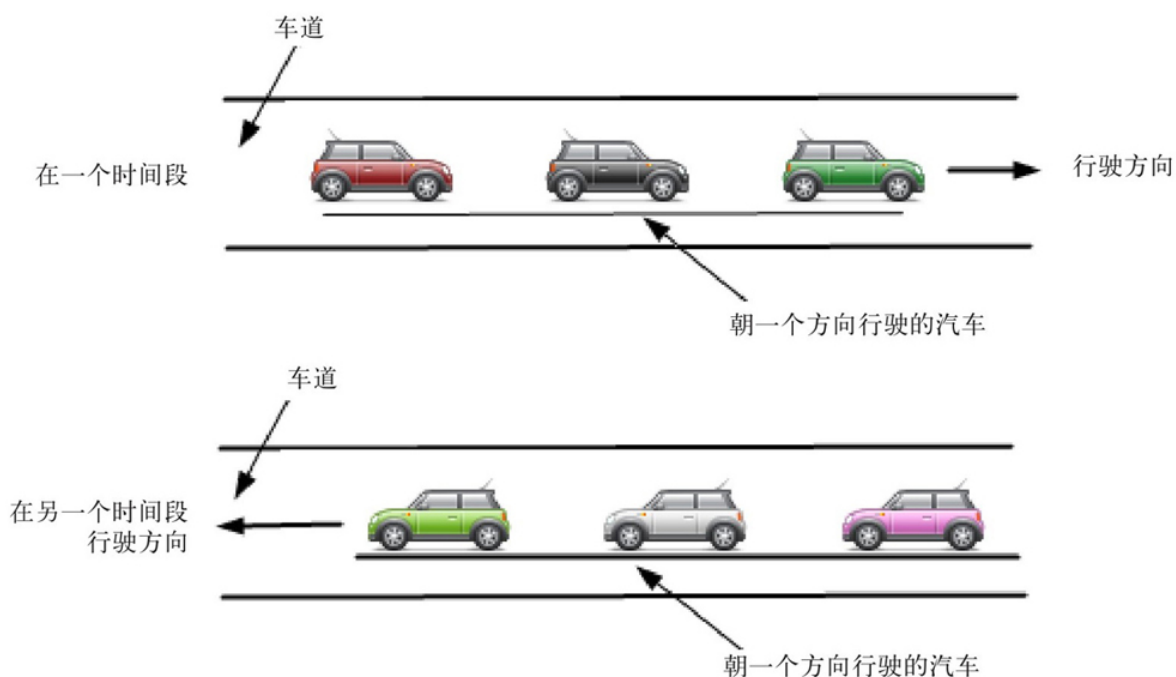


图 4-20 半双工通信类比示例

3.全双工通信

图4-18c所示为全双工通信方式。此时信号任何时刻都可以同时双向传送。它要求至少存在两条信道。也就是说，“全双工”在同一时刻可以进行两个方向的数据传输，而且互不影响。这就相当于我们现实生活中的“双向双车道”。在这样的公路中，两个方向的汽车都可以同时通过，如图4-21所示。很明显，这样可以提高汽车通过的效率。对于数据通信来说，全双工的线路的传输速率就是提高了数据传输速率，

最高可以达到带宽的两倍。如100Mbps的全双工线路，最终的数据传输速率最高可达200Mbps。这也就是4个100Mbps端口聚合而成的以太网通道，最高可以达到800Mbps的数据传输速率的原因了。

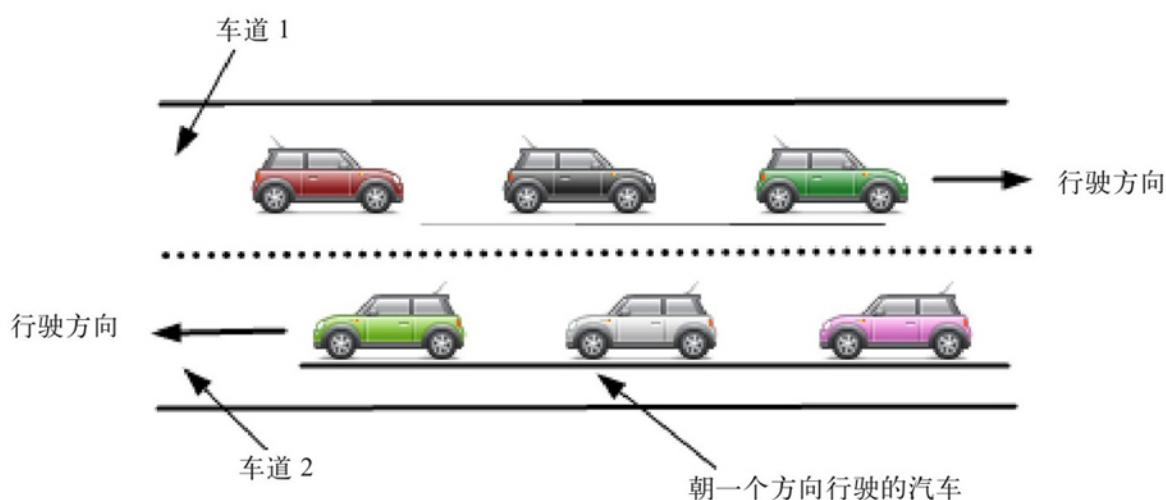


图 4-21 全双工通信类比示例

一个以太网端口的工作模式有：10Mbps半双工、10Mbps全双工、100Mbps半双工、100Mbps全双工、1000Mbps全双工和自动协商这么几种。当然最终能支持哪几种，要根据设备上各端口所支持的网络技术标准而定。

一般来说，建议将通信双方的端口工作模式设置一致，否则要么出现不能成功连接，要么会降低另一端的端口数据处理性能。如一端设置为单工，而另一端设置为半双工，则最终只能选择最低要求的半双工模式进行通信。现以普通的10/100Mbps自适应快速以太网端口为例进行介绍，具体规则如下：

□如果一端是固定模式（无论是10Mbps还是100Mbps），另外一端是自协商模式，即便能够协商成功，自协商的那一端也只能工作在半双工模式。

□如果一端工作在全双工模式，另外一端工作在半双工模式（包括自协商出来的半双工），Ping是没有问题的，流量小的时候没有任何问题，流量达到约15%以上时，就会出现冲突、错包，最终影响工作性能。

□对于两端工作模式都是自动协商，最后协商成的结果是“两端都支持的工作模式中优先级最高的那一类”，通常为半双工模式优先。

□如果A端为自动协商模式，B端设置为100Mbps全双工模式，则在A协商为100Mbps半双工模式后，再强制将B改为10Mbps全双工模式，A端也会马上向下协商到10Mbps半双工模式；如果A端为自动协商模式，B端设置为10Mbps全双工模式，则在A协商为10Mbps半双工模式后，再强制将B改为100Mbps全双工模式，则会出现协商不成功，连接不上的现象。这个时候，如果插拔一下网线，又会重新协商为100Mbps半双工模式。

4.3 数据传输速率与信道带宽

在数据通信中，数据传输速率与信道带宽（或称传输带宽）是两个非常重要的概念。它们定义了传输链路上的物理层基本特性，而且两者之间存在一些必然的联系，本节要详细介绍数据传输速率与信道带宽的计算，以及它们之间的关系。

4.3.1 传输速率与信道带宽的基本概念

在本章前面已介绍到了，传输速率是在物理层的电气特性中定义的。“数据传输率”是指在一条信道中，单位时间内传输的信息量，与我们平时描述河流水速是一样的。数据传输速率可用比特率（针对数字数据）或波特率（针对模拟数据）来表示。

（1）比特率

比特率通常又称信息传输速率，是指单位时间内传输的二进制代码的有效位（bit）数，用 R_b （注意这里是小写的b）表示。比特率是用来描述可直接在计算机网络数字线路上传输的数字数据的传输速率，常用的单位有bit/s（b/s或bps，每秒比特数）、kbit/s（kb/s或kbps，每秒千比特数）或Mbit/s（Mb/s或Mbps，每秒兆比特数）。但

要注意，此处的k和M分别为1000和1 000 000，而不是涉及计算机存储器容量时的1024和1048576。

根据以上“比特率”的定义，可用以下公式来计算信息传输速率。

$$R_b = 1/T \text{ (b/s)} \quad (4-1)$$

其中， R_b 为信息传输速率， T 为发送每一比特数据所需要的时间。例如，如果在通信信道上发送1位0或1信号所需要的时间是0.001ms，那么根据公式可计算出信道的数据传输速率为1 000 000bps=1Mbps。

(2) 波特率

波特率 (Baud rate) 是指一个数字信号在被调制后，数字信号对载波的调制速率，也即单位时间内载波参数（如频率、相位等）变化的次数，单位为B。如果是非调制的二进制数字信号，波特率与比特率数值上是相等的，所以波特率通常是针对经过调制后的数字信号传输速率。

波特率的单位为Baud，用 R_B （注意这里是大写的B）表示。一个数字脉冲，或者直接说是一个二进制位称为一个码元（如图4-9所示），波特率也可以理解为在单位时间内传输的码元数。这里的码元可以是二进制的，也可以是多进制的。如二进制的字母A的ASCII码是

01000001，可用7个脉冲来表示（最高位为校验位，不计入码元中），所以A可认为是由7个码元组成的。若1s内传2400个码元，则波特率为2400B。

（3）比特率与波特率的关系

对于数字数据，比特率与波特率的关系与数据所采用的进制有关。因为每个码元或符号通常都含有一定比特数的信息量，所以比特率（ R_b ）与波特率（ R_B ）有如下关系：

$$R_b = R_B \times \log_2 M(\text{b/s}) \quad (4-2)$$

式中的M为符号的进制数，如二进制、八进制、十六进制。例如码元速率为1200B，采用八进制（ $M=8$ ）时，则可计算出对应的信息速率 $R_b = 1200 \times \log_2 8 = 3600(\text{b/s})$ ；如果采用的是二进制（ $M=2$ ），则信息速率为 $R_b = 1200 \times \log_2 2 = 1200(\text{b/s})$ ，可见，二进制的波特率和比特率在数值上是相等的。

在采用了数字调制技术的数字通信中，比特率与波特率的关系还与所采用的调制技术有关。因为具体的调制技术将在本章后面专门介绍，所以在此仅给出它们的关系式，关系式与式（4-2）一样。不过，此时M代表的是调制的相位数。

两相调制（单个调制状态对应1个二进制位）的比特率的数值等于波特率的数值；四相调制（单个调制状态对应2个二进制位）的比特率的数值为波特率的数值的2倍；八相调制（单个调制状态对应3个二进制位）的比特率的数值为波特率的数值的3倍；依此类推。

（4）传输带宽

“带宽”是指信道中每秒传输的最大信息量，也就是一个信道的最大数据传输速率，单位也是“位/秒”（b/s或bps）。带宽是一种理想状态（不受任何干扰，没有任何衰减）下的信道传输速率，而上节介绍的“数据传输速率”是指当前实际的数据传输速率，通常情况下“数据传输速率”永远小于或等于“带宽”值（事实上一般不可能等于带宽的，因为传输过程中不可能完全不受干扰，线路性能一般也不可能完全达到理想状态）。可以打这样一个比方，如一条河流它能承受的最大水流速度为 $1000\text{m}^3/\text{s}$ ，但现在由于河里的水不多，实际的水流速度仅 $100\text{m}^3/\text{s}$ 。这就是传输带宽与数据传输速率的关系了。

4.3.2 数字信号不失真传输的最大传输速率限制

物理层的任何一个信道，它所能承受的信号频率（或者说是信道带宽）是有限的，就像任何一条河流只能承受一定峰值的水流量一样。在信道中，超出限定的频率范围的信号都将被直接滤出，不能通过信道，就像河流中任何超出它流速限制的水流量最终不能形成实际速率的水流量一样。

在数据通信中有数字信号传输和模拟信号传输之分（对应也就有数字信道和模拟信道之分了），但数字信号的传输要求与模拟信号的要求不同：模拟信号的传输要求接收端无波形失真，而数字信号的传输则要求接收端无差错地恢复成原来的二进数码（可以允许接收波形失真，只要不影响正确恢复信号编码即可）。在模拟信号传输中，为了达到不失真的效果，就需要对由模拟信号转换成数字信号过程中的采样频率进行限制；而在数字信号传输中，就需要对信号传输速率进行限制。本节仅介绍数字信号传输中信号传输速率的限制。

在计算数字信道中最大传输速率方面，有两个非常有名的准则或定理，那就是奈奎斯特准则（简称奈氏准则）和香农公式。

1. 奈奎斯特准则

由于数字信号（这里特指没有被调制的原始数字信号，也就是上面所说的数字基带信号）的频带（这里是指信号的频率带宽）可以非常宽（从直流一直到无限高的频率），但其主要能量集中在低频段，而且数据通信中的电缆传输信道只允许比较低的频率成分通过理想低通信道，高频成分将被滤去，这就造成了输出波形的失真。失真的输出波形顶部变圆，底部变宽（称为波形拖尾），使得一个码元的波形展宽到其他码元位置，影响到其他码元，这种影响称为码间干扰。

出现码间干扰后，波形的拖尾可以是正，也可以是负：如果某个码元内所有的拖尾部分相加后是正值，而且达到门限判决电平（也就是判断信号电平为1或为0的基准电平）就可能将“0”误判为“1”码；反之，如果所有的拖尾相加后在某个码元内的值是负的，就可能将“1”码误判为“0”码。

说明 理想“低通信道”是指只要信号频率不超过某个上限值都能够不失真地通过此信道；而频率超过该上限值的所有高频分量都不能通过该信道。与之相对的还有一种称为理想“带通矩形信道”，它是指仅允许在上下限之间的信号频率成分不失真地通过，其他频率成分不能通过。

要使信道中传输的数字信号不失真，信道中的数据传输速率必须限制在某个范围之内，这又与信道类型有关。1924年，美国著名物理学家奈奎斯特（Nyquist），经过多次实验证明，为了确保数字信号不

失真传输，在理想低通信道（也就是在4.1.4节介绍的基带传输中的信道）下实际的最高码元传输速率（即无码间干扰的最高波特率）必须在下式计算结果之内：

$$\text{最高码元传输速率 (MaxRB)} = 2W \quad (4-3)$$

式中W是理想低通信道的带宽，单位为赫（Hz），MaxRB为最高码元传输速率，单位为Raud（波特）。这就是著名的奈奎斯特第一准则，其指出了在理想低通信道（也就是4.2.4节中介绍的基带传输中的信道）中，为了确保数字信号的不失真传输，信道带宽和最大码元传输速率之间的关系。从该公式可以得出，每赫兹带宽的理想低通信道的最高码元传输速率是每秒2个码元。若码元的传输速率超过了这一准则所给出的数值则将出现码间干扰，以致在接收端无法正确判定码元是1还是0。

说明 奈奎斯特是美国的一位著名物理学家。他1889年出生于瑞典，1907年移民到美国并于1912年进入北达克塔大学学习，1917年在耶鲁大学获得物理学博士学位，1917年~1934年在AT&T公司工作，后转入贝尔电话实验室工作，最终于1976年在德克萨斯逝世。奈奎斯特对信息论做出了重大的贡献。

例如，低通信道带宽为2000Hz时，最高码元传输速率就为4000 Baud，即每秒最多可传送4000个二进制码元。反过来，根据最大码元

传输速率也可算出信道带宽。如一路数字电话速率为64kbps（因为是数字电话，采用的是二进制，所以比特率的数值与波特率的数值是相等的，所以此电话的波特率为64000 Baud），则根据以上公式可计算出无码间干扰的信道带宽为32kHz。

另外，在奈奎斯特第一准则中也同时得出，在理想带通矩形信道（也就是在4.2.4节介绍的频带传输中的信道）下的最高码元传输速率与信道带宽是具有相等值关系的，即每赫兹宽带的带通信道的最高码元传输速率为每秒1个码元。

2.香农公式

以上奈奎斯特第一准则描述了有限带宽、无噪声信道的最大数据传输速率与信道带宽的关系。美国的另一位著名物理学家——克劳德·香农(Claude Elwood Shannon)也在数据通信领域有权威的发现，那就是“香农公式”。这个公式描述了有限信道带宽、有随机热噪声信道（更接近真实环境）的最大传输速率与信道带宽、信噪比（信号平均功率与噪声功率之比）之间的关系。香农公式指出：在有随机热噪声的信道上传输数据信号时，最大数据传输速率 R_{\max} 与信道带宽 B 、信噪比 S/N 的关系如下所示：

$$R_{\max}=B \times \log_2 (1+S/N) \quad (4-4)$$

式中 R_{\max} 的单位为bps，信道带宽 B 的单位为Hz，信噪比 S/N 通常用dB（分贝）表示，可用 $10 \times \log_{10} (S/N)$ 公式来计算信噪比的分贝数。若已知 $S/N=1000$ ，则根据公式可很快得出用分贝数表示的信噪比为30dB。如果再已知信道带宽 $B=3000\text{Hz}$ ，则 $R_{\max} \approx 30\text{kbps}$ 。它表示对于带宽只有3000Hz的通信信道，信噪比在30dB时，无论数据采用二进制或更多的离散电平值表示，都不能用越过30kbps的数据传输速率。

4.3.3 模拟信号不失真还原的最小采样频率限制

上节介绍了数字信号不失真传输时，信道中的最大传输速率限制，本节要介绍的是模拟信号传输中在接收端不失真还原的最小采样频率限制。

我们知道，在计算机设备中只能接收数字信号，如果在信道中传输的是模拟信号（通过公共交换电话网络进行的远程网络连接的电话线路中传输的都是模拟信号），在接收端必须进行模/数转换，生成数字信号；同样如果将在一个计算机设备中的数字信号还原到某条模拟线路中进行传输，则又要进行数/模转换，还原成原来的模拟信号。这里最关键的是由模拟信号转换成数字信号，因为在这个转换过程中要确保数据不失真。

因为模拟信号是随时间连续变化的，而数字信号则是离散的，所以在模/数转换过程中，就需要离散地在模拟信号上抽取一定的信号样本，形成数字信号样本，以便可以不失真地代表原始数据本身，也使得抽取的数字信号样本在需要时能还原为原始的模拟信号。这就涉及一个“采样频率”（简单地说就是每隔多少时间提取一次信号波）的问题。

从信号处理的角度来看，采样定理描述了两个过程：其一是采样，这一过程将连续时间信号转换为离散时间信号；其二是信号的重建，这一过程是将离散信号还原成连续信号。连续信号在时间（或空间）上以某种方式变化着，而采样过程则是在时间（或空间）上，以 T 为单位间隔来测量连续信号的值。 T 称为采样间隔。在实际中，如果信号是时间的函数，通常它们的采样间隔都很小，一般在毫秒、微秒的量级。采样过程产生的一系列的数字称为样本，代表了原来的信号。每一个样本都对应着测量这一样本的特定时间点，而采样间隔的倒数，即 $1/T$ 表示采样频率（ f_s ），其单位为样本/秒，即赫（Hz）。信号的重建是对样本进行插值的过程，即从离散的样本 $x[n]$ 中，用数学的方法确定连续信号 $x(t)$ 。

要确保信号不失真的恢复，最低的采样率需要达到多少呢？采样过程所应遵循的规律称为采样定理，又称取样定理、抽样定理。采样定理说明采样频率与信号频谱之间的关系，是连续信号离散化的基本依据。采样定理于1928年由美国物理学家奈奎斯特首先提出来，因此称为奈奎斯特采样定理。1933年苏联工程师科捷利尼科夫首次用公式严格地表述这一定理，因此在苏联文献中称为科捷利尼科夫采样定理。1948年信息论的创始人香农对这一定理加以明确说明并正式作为定理引用，因此在许多文献中又称为香农采样定理。

上述采样定理又有许多表述形式，但最基本的表述方式是“时域采样定理”和“频域采样定理”。在计算机网络的数据通信中，主要用到的是“时域采样定理”。

时域采样定理中规定，频带为 F 的连续信号 $f(t)$ 可用一系列离散的采样值 $f(t_1)$, $f(t_1 \pm \Delta t)$, $f(t_1 \pm 2\Delta t)$,来表示，只要这些采样点的时间间隔 $\Delta t \leq 1/2F$ ，便可根据各采样值完全恢复原来的信号 $f(t)$ 。简单地讲就是，在进行模拟/数字信号的转换过程中，当采样频率 $f_{s, \max}$ 大于或等于信号中最高频率 f_{\max} 的2倍时($f_{s, \max} \geq 2f_{\max}$)，采样之后的数字信号就可完整地保留原始信号中的信息，原来的连续信号就可以从采样样本中完全重建出来，否则采样信号后的频率将会混叠。一般实际应用中保证采样频率为信号最高频率的5~10倍。

由以上定理可以得知，如果已知信号的最高频率 f_H ，就可以得出保证完全重建信号的最低采样频率是 $2f_H$ 。相反，如果已知采样频率，则可以得出保证信号完全重建所允许的最高信号频率。但要注意的是，这里有一个条件，那就是被采样的信号必须是带限的，也就是是有限带宽的，信号中高于某一给定频率的成分必须是0，或者至少非常接近0。只有这样，在重建信号时这些频率成分的影响才可以忽略不计。如果被采样的信号不是带限的，采样后信号的频率就会产生重叠，即高于采样频率一半的成分将被重建成低于采样频率一半的信号，最终导致信号不能被完全恢复。

频域采样定理中规定了对于时间上受限制的连续信号 $f(t)$ 的采样频率。因为比较复杂，且在数据通信中比较少用，故在此不再介绍。

4.4 数字基带信号编码

在本章前面我们提到了，数字基带传输模式中传输的信号是要经过编码的原始数字信号。但数字基带信号的类型有很多，常见的有矩形脉冲、三角波、高斯脉冲和余弦脉冲等。其中最常用的是矩形脉冲，因为矩形脉冲易于形成和变换。本节先介绍矩形脉冲数字信号中几种常见的波形和编码方式。

4.4.1 矩形脉冲数字信号基本波形

在矩形脉冲数字信号中主要有以下四种波形：单极性不归零波形、单极性归零波形、双极性不归零波形、双极性归零波形。

1. 单极性波形与双极性波形

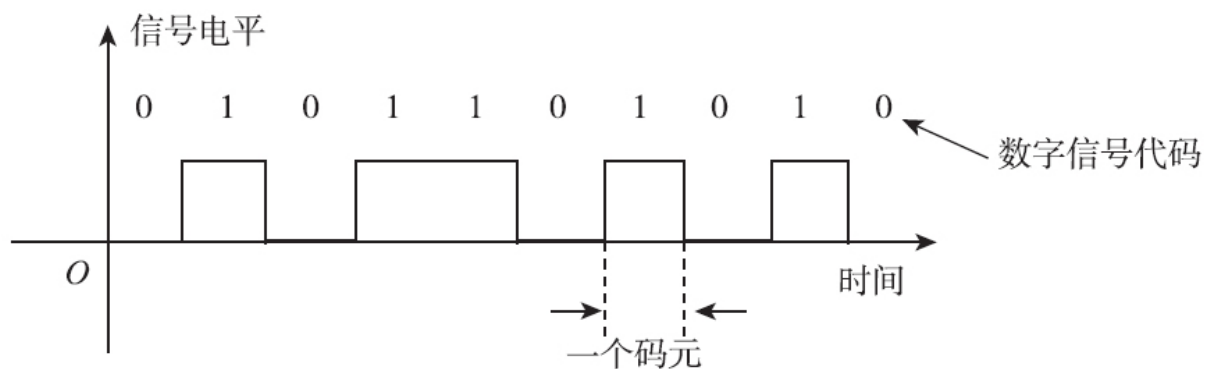
在以上四种矩形脉冲波形中首先要理解两个概念，那就是什么是“单极性”，什么是“双极性”。

“单极性”就是波形中仅用正（或负）电平值来表示信号中的二进制“1”（通常是以正电平表示），零电平表示信号中的二进制“0”。对应电平信号是1还是0（也就是通常所说的“判决电平”）以信号电平矩形脉冲幅度（即信号波的“振幅”）的 $1/2$ 为基准，信号电平值大于或等于

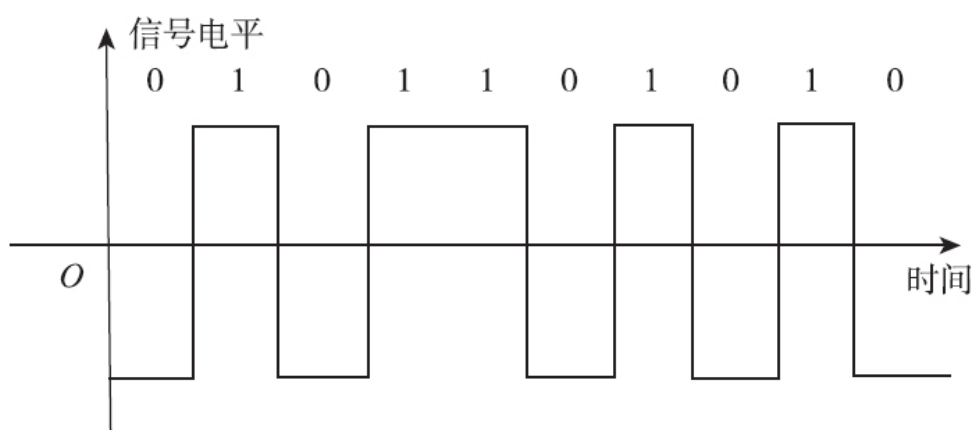
振幅的 $1/2$ 时就用1表示，小于振幅的 $1/2$ 时就用0表示。之所以称之为“单极性”，就是因为它仅使用了其中一个极性（要么用正，要么用负，一般用正极性）。如图4-22a所示的就是一个信号代码为0101101010的单极性矩形脉冲波。

而“双极性”是指分别用正和负电平值表示信号中的二进制“1”和“0”，而且正、负脉冲幅度一样。此时的判决电平就是零电平，信号电平值大于0时用1表示，信号电平值小于0时用0表示。之所以称之为“双极性”就是因为它同时使用了正、负两种极性。

如图4-22b所示的是一个信号代码为0101101010的双极性矩形脉冲波。



a) 单极性矩形脉冲波形



b) 双极性矩形脉冲波形

图 4-22 单极性矩形脉冲波形和双极性矩形脉冲波形

以上两种波形都是在一个码元的全部时间内发出或不发出电流（仅适用于单极性波形），以及发出正电流或负电流（仅适用于双极性波形）。每一位编码占用整个码元的宽度，所以这两种编码都属于“全宽码”。但这样的波形存在一个码元难以识别的问题，因为当连续发送1码，或者0码时，就会使某一位码元与其下一位码元之间没有间隙（因为持续有电流），容易使接收方误认为对应的信号电平是稳定的直流电平。正因如此，在数据通信中一般不是直接采用这种原始

波形进行传输的，而是采用了归零码这种编码方式，而原来这种不重新编码的波形称为非归零码。非归零码在传输中难以确定一位的结束和另一位开始，需要用某种方法对发送器和接收器进行定时或同步。

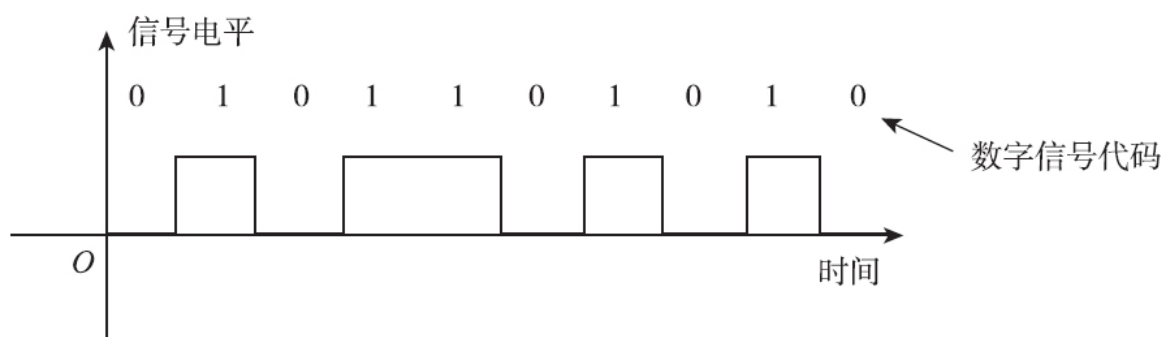
2.归零码与非归零码

理解什么是归零码和非归零的关键就是要理解什么是归零。归零是指信号电平在一个码元宽度内（通常是在 $1/2$ 个码元时）信号脉冲电平必须回归为零电平（也就是无电流），直到该码元宽度结束。非归零码就是不对脉冲信号波进行任何编码，原来是什么样子，传输的就是什么样子。

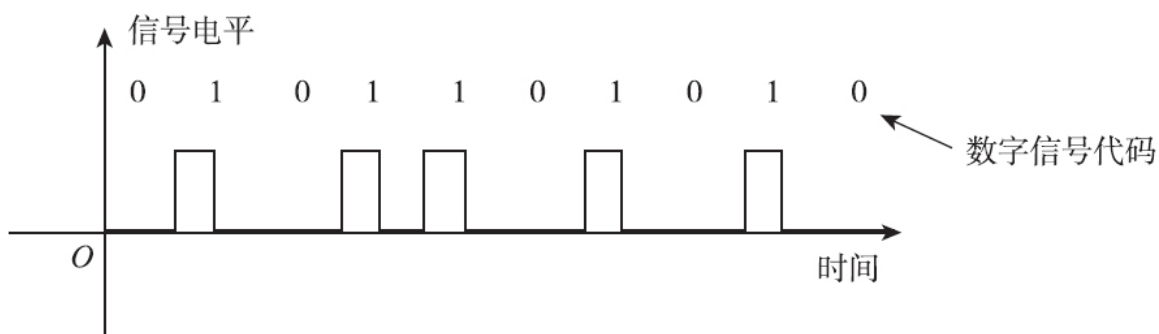
因为矩形脉冲波有单极性和双极性之分，归零码对应有单极性归零码和双极性归零码，非归零码对应有单极性非归零码和双极性非归零码。一般来说，直接采用单极性非归零码和双极性非归零码比较少。单极性非归零码主要适用于极短距离传输，更多是需要再进行其他形式的编码，如下节将要介绍的各种传输码，都是在单极性非归零码基础上改进的，最终都变成了双极性的编码。双极性非归零码常在CCITT的V系列接口标准或RS-232接口标准中使用。

从以上对归零这两个字的理解，我们可以很快理解单极性归零码的含义。因为单极性非归零码仅采用一个极性，并且“1”码表示有信号

(是正电平还是负电平，取决于所选择的极性)， “0”码表示无信号(零电平)， 所以把单极性非归零码转换成单极性归零码就很简单了， 只需要把“1”码的脉冲电平在其码元宽度内(通常是在1/2个码元时)从正电平或负电平回归到零电平， 一直持续到该码元结束， 原来为“0”码的保持零电平不变， 因为它已是零电平了， 再回归零电平还是零电平。图4-23所示的就是一个单极性非归零码转换为单极性归零码的示例。



a) 单极性非归零码

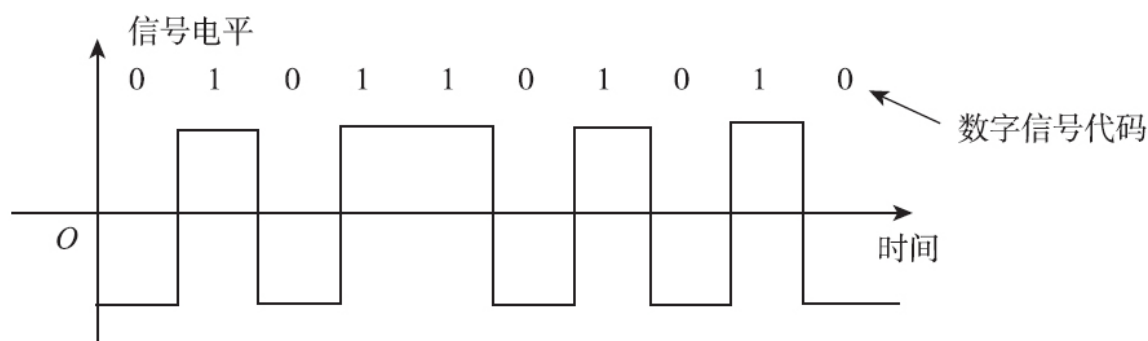


b) 单极性归零码

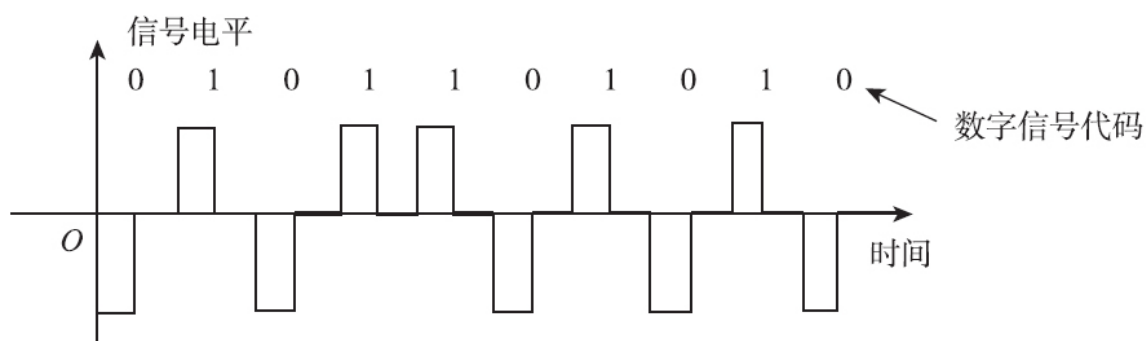
图 4-23 单极性非归零码转换为单极性归零码的示例

从以上叙述可以看出，单极性归零码的主要优点是可以直接提取同步信号，因为在出现连续的1码元时，都会有一个小段（通常为1/2个码元宽度）的零电平间隙，不会出现任何两个码元连续的正电平（或负电平），比较容易区分有信号时两个码元之间的界限。但它仍有一个不足，就是当下一个码元也是无信号的零电平时，则两个码元仍会是连续的零电平，仍比较难实现数据传输的同步，因为仍不能区分两个连续0码元的界限。而且单极性码中采用了零电平，包括了直流和低频成分，不利于在信道中传输。

由双极性非归零码转换成双极性归零码的原则的理解，主要也是对“归零”两个字的理解，具体的理解方式同上，只不过，因为双极性非归零码中的信号不是采用零电平，而是采用正电平来表示1码元，负电平表示0码元，所以此时的双极性归零码中的两个极性（不管是正电平，还是负电平）的码元电平都必须在一个码元宽度内（通常也是在1/2个码元时）回归零电平，一直持续到该码元结束。图4-24所示的就是一个双极性非归零码转换为双极性归零码的示例。



a) 双极性非归零码



b) 双极性归零码

图 4-24 双极性非归零码转换为双极性归零码的示例

从以上叙述可以看出，在双极性归零码中永远不会出现任何两个码元电平的连续，不管是正电平，还是负电平，在其码元宽度间（通常为 $1/2$ 个码元宽度）都会回归到零电平，每个码元之间都有一个小段（通常为 $1/2$ 个码元宽度）零电平的间隙，非常容易区分不同码元，也非常容易实现数据的同步传输。另外，尽管双极性码编码前不采用零电平，仅有正、负电平，但在编码后仍含有较多的零电平直流成分，所以也不宜在信道中传输。

如果把图4-22~图4-24所示内容合并在一起，就更容易通过对比看出它们之间的区别，以及非归零码转换成归零码的规则了，如图4-25所示（它们传输的信号代码相同）。

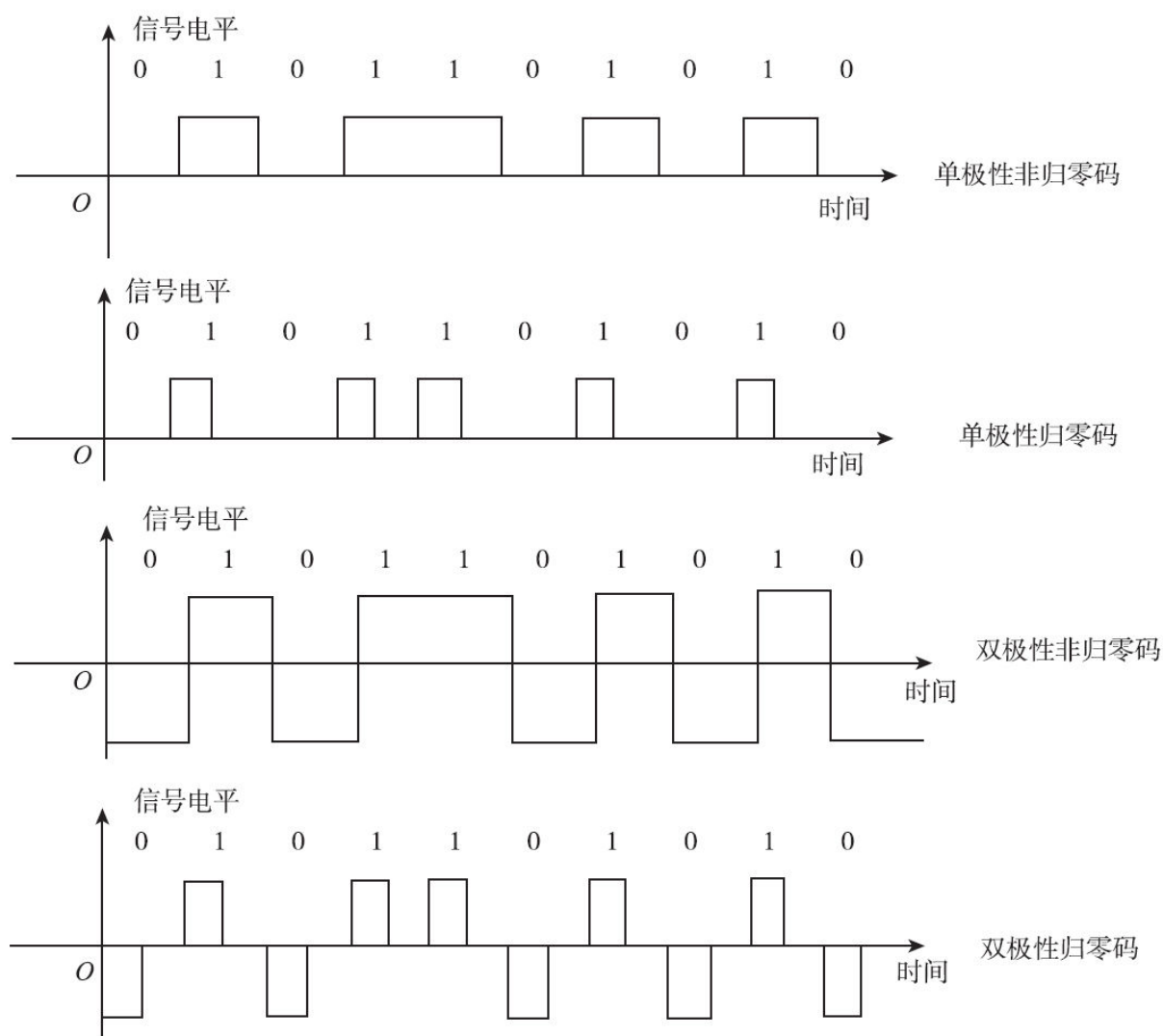


图 4-25 单/双极性非归零码和单/双极性归零码的比较

4.4.2 数字基带信号的传输码型

在实际的基带传输系统中，并不是所有编码类型的电信号波形都能在信道中传输。例如上节介绍的含有直流分量和较丰富低频分量的单极性基带波形就不适宜在低频传输特性差的信道中传输，因为其功率较弱，在传输过程中有可能造成信号严重畸变。另外，当信息代码中包含有长串连续“1”或“0”时，非归零波形呈现出连续的固定电平，会出现无法获取同步的问题。单极性归零码在传送连续“0”信息代码时也会存在同样的问题。因此，信息码在进行传输之前，必须经过码型变换，变换为适用于信道传输的码型，这就是通常所说的传输码，也称线路码或信道编码。

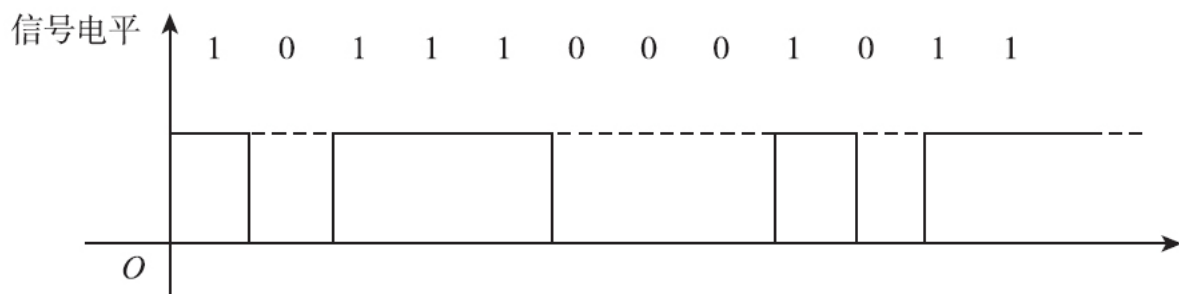
传输码型的选择原则：一是要能从基带信号中提取定时信息（位同步脉冲信息），所以要求传输码型应含有（或者经变换后含有）时钟频率分量（不能出现0电平），且编码转换后不能出现过多的连续“0”码或“1”码，否则提取的时钟信号就会很不稳定，引起同步偏移；二是要无直流分量、低频成分少，否则影响信号在信道中的传输；三是信号中高频分量要尽量少，以节省传输频带并减少码间串扰。

常见的传输码型有：传号反转交替码（AMI码）、三阶高密度双极性码（HDB3码）、传号反转码（CMI码）、数字双相码（曼彻斯特

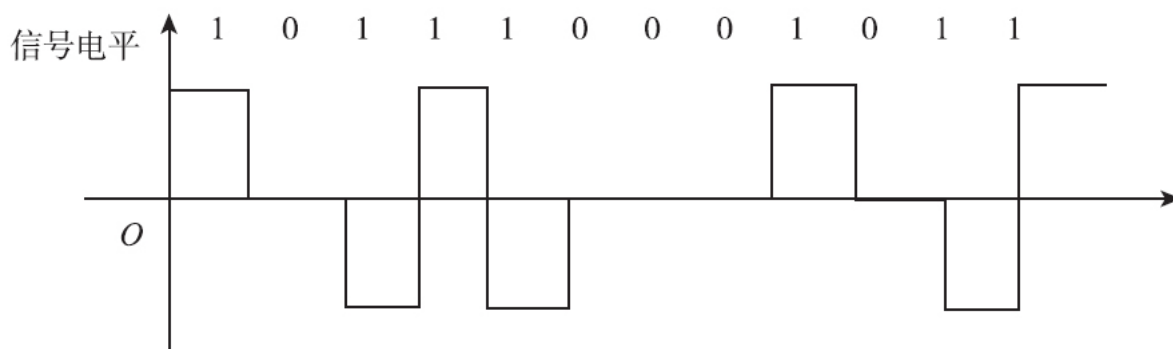
码) 和差分曼彻斯特码。上节介绍的两种归零码也是传输码类型。下面分别介绍。

1.AMI码

AMI (Alternative Mark Inversion) 码是对单极性非归零码的变形(也称非归零AMI码), 其编码规则是将单极性非归零码的二进制信号代码中的“1”码元交替地替换为正、负电平(可以用+1、-1来理解), 而“0”码元保持不变。这里的“传号反转交替”就是指信号中“1”码转换后变成正、负极性交替(但相邻“1”码的极性变换方向相反)的代码。如一个单极性非归零码中的二进制信号代码为1 0 1 1 1 0 0 0 1 0 1 1, 转换成AMI码后就为+1 0 -1 +1 -1 0 0 0 +1 0 -1 +1, 对的波形如图4-26所示。可以看得出, 其编码规则也很简单。



a) 单极性非归零码



b) AMI码

图 4-26 AMI码与单极性非归零码的比较

AMI码的优点是：由于“1”码元采用了正、负电平交替，所以AMI码的有信号部分的频谱中不含直流成分，高频和低频成分也很少。另外，AMI码的编译码电路简单，便于利用传号极性交替规律观察误码情况。鉴于这些优点，AMI码是CCITT建议采用的传输码型之一。

AMI码的不足是：当原二进制信号代码出现连续的“0”码时，信号电平会长时间不变，造成提取定时信号的困难（也就是同步比较困难），解决连续“0”码问题的有效方法之一是采用后面要介绍的HDB3码，它是AMI码的改进码型。

2.HDB3码

为了利用AMI码的优点，并克服其缺点，科学家就提出了许多种类的改进AMI码。其中的典型代表就是各种高密度双极性码HDBn（所以HDBn码也是对单极性非归零码的变形码），HDB3码（High Density Bipolar of order 3 code）就是高密度双极性码集中最典型的一种，其最关键的特性就是可以确保信号波形中连续零电平（其实也就是码元连续为0的个数）的码元数不超过3个。具体的编码步骤如下：

1) 先把基带数字信号代码变换成AMI码，然后检查AMI码中的连续“0”码元的情况，如果信号码中的连续“0”码的个数不超过3，则这时的HDB 3码就是原来的AMI码。如一个二进制信号代码为1 0 0 1 1 0 1 1 0，转换成的AMI码为+1 0 0 -1 +1 0 -1 +1 0，从中可以看出连续“0”码最多只有2个，没有超过3个，所以该信号的最终HDB3码与其AMI码一样。

2) 当信号AMI码中出现连续4个或4个以上“0”码元时，则将每4个（就是每4个一组）连续“0”码元的第4个“0”转换成非0脉冲，记为+V或-V（是取“破坏”之意的Violation单词的第一个字母），称之为“破坏脉冲”。原来的信号码元序列中所有的“1”码称为“信码”，用符号B表示。至于第4个“0”码是转换成+V码，还是-V码，取决于以下两个原则：①V码必须与前一个信码B极性相同（同时为正，或同时为负）；②相邻V码（包括+V码和-V码）的极性必须相反。

图4-27所示的是10000100001信号代码的单极性非归零码、AMI码和HDB3码的比较。在第一个连续4个0码段中，把第4个0码的脉冲变为V脉冲，因为它前一个信码B（也是前一个“1”码）的极性为正，所以它的极性也为正。在第二个连续4个0码段中，把第4个0码的脉冲也变为V脉冲，但因为它前一个信码B的极性为负，所以它的极性也为负。最终得到的HDB3码就为+1000+V-1000-V0+1。

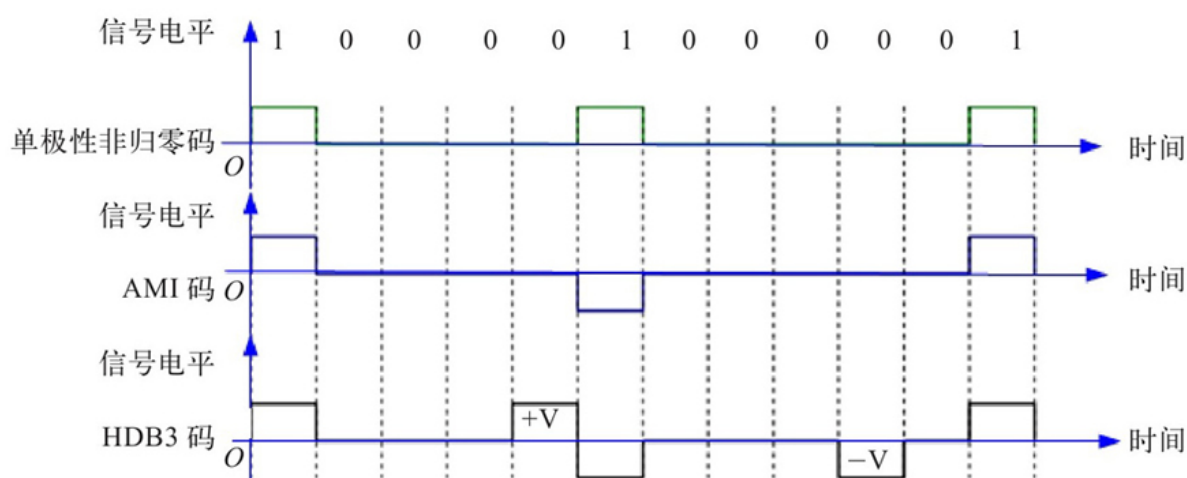


图 4-27 单极性非归零码、AMI码和HDB3码的比较示例

3) 如果通过变换V码后，得到的波形还不能同时满足以上两个原则，则需要将对应的4个连续“0”码的第一个“0”码转换成与该段4个连续“0”码的V码同极性的补信码，用符号B'表示。最终使得V码与前一个信码（包括B码和B'码）的极性相同，相邻V码的极性以及相邻信码的极性都是相反的。

如现在信号代码为1 0 0 0 0 1 1 0 0 0 0 1，如果仅把第4个“0”码转换成对应极性的V码，则得到如图4-28所示第3个波形的不正确HDB3码。因为此时第二个V码的极性为负，与它前面一个B码极性不相同，不满足第2步中说到的第①个原则。此时就要将第二个连续4个0码的码段第一个“0”转换为与前一个B码极性相反的B'码。如图4-28所示的第4个波形即为正确HDB3码。这时就全部符合以上规则了。最终的HDB3码就为：+1 0 0 0 +V -1 +1 -B' 0 0 -V -1。

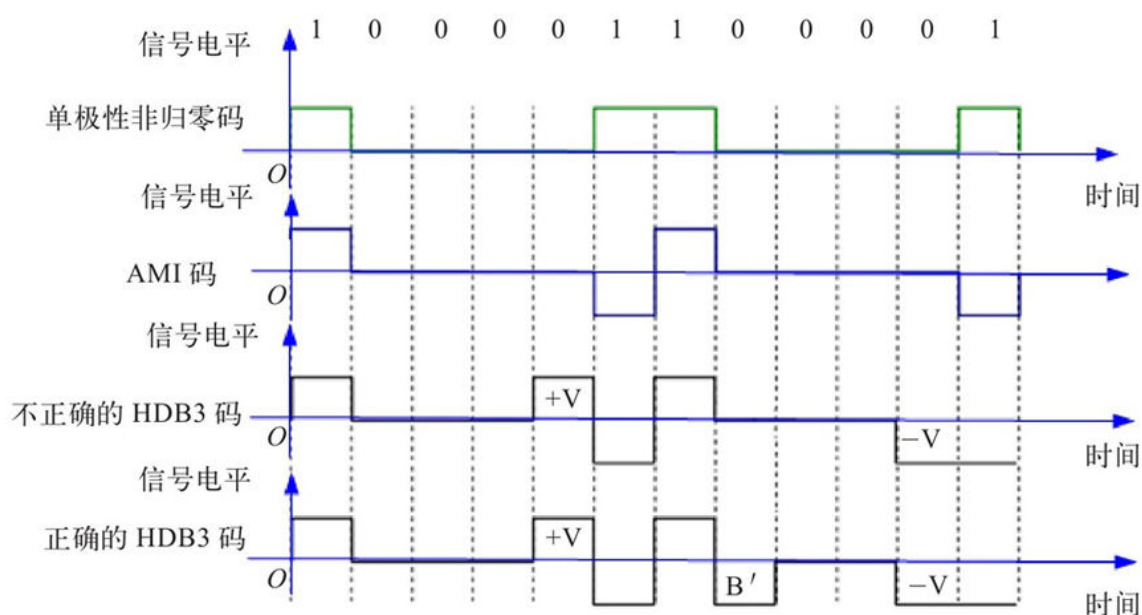


图 4-28 加B'码的示例

HDB3码除保持了前面介绍的AMI码的优点外，同时还将连续“0”码限制在3个以内，更有利于位定时信号的提取。HDB3码是应用最为广泛的码型，已成为CCITT协会推荐使用的基带传输码型之一。

3.CMI码

CMI (Coded Mark Inversion) 是一种双极性二电平非归零码, 又称1B2B码 (即一位信息码用二位表示)。所谓“二电平”就是一个码用两个电平来表示。通过上节双极性码的介绍我们知道, “1”码用正电平表示, “0”码用负电平表示, 电平判决值就是零电平, 也就是大于零电平的码为“1”, 小于零电平的码为“0”。

CMI码的编码规则也比较简单, 就是把基带数字信号中的“1”码交替用正、负电平表示 (也就是如果前一个“1”码为正电平的话, 后一个“1”码则用负电平表示), 得到的CMI码对应为“11”和“00” (两者交替出现, 两个电平各占半个码元宽度); “0”码固定用1/2码元宽度的负电平和1/2码元宽度的正电平表示 (注意前后1/2码的极性是固定的), 得到的CMI码恒定为“01”。简单地说就是: 当传送一个“0”码时, 编码后输出固定的“01”码; 当传送一个“1”码时, 编码后交替输出“11”或者“00”码。

图4-29所示的是基带数字信号110001110的单极性非归零码和CMI码的比较。最终得到的CMI码为110001010111001101。从图中可以看出, 原来信号中的每一位都变成了两位。

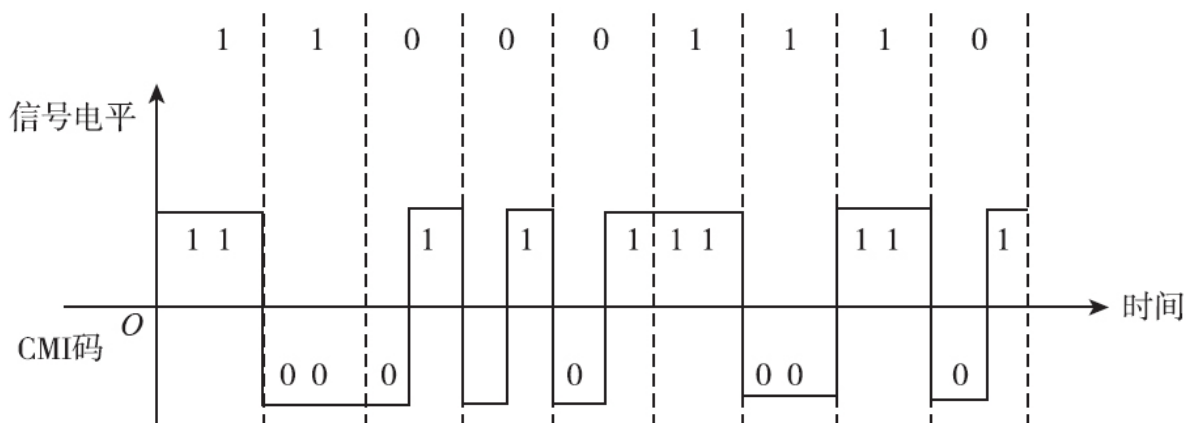


图 4-29 CMI码转换示例

CMI码具有一定的检测错误的能力，因为“1”码交替地用“11”和“00”来表示，而“0”又固定用“01”表示，所以正常情况下，转换后的编码“10”是不可能出现在信道中的，连续的“00”或“11”也是不可能出现的，这样一来就可以用来检测因信道传输而产生的错误。但是CMI码不能纠错。CMI码也没有直流分量（没有零电平，只有正、负电平），且有频繁出现的波形跳变，便于恢复定时信号（也就是便于同步）。

4.曼彻斯特码

曼彻斯特码（Manchester Encoding）与上面介绍的CMI码类似，它也是一种双极性二电平非归零码。其编码规则是：每个原始信号码元固定用两个连续极性相反的脉冲来表示，“1”码用正、负电平表示（也就是先正后负，对应的编码为“10”），“0”码用负、正电平表示（也就

是先负后正，对应的编码为“01”）。其关键特点就是在每一码元的1/2码元位置必须进行极性跳变。

图4-30所示的是基带数字信号110001110的单极性非归零码和曼彻斯特码的比较。最终得到的曼彻斯特码为1010010101101001。从图中可以看出，原来信号中的每一位都变成了两位。

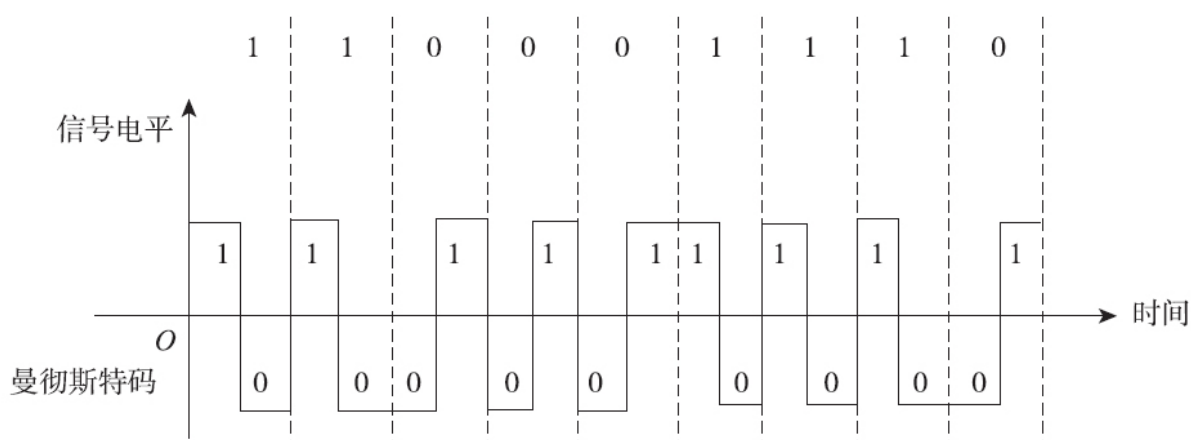


图 4-30 曼彻斯特码转换示例

曼彻斯特码的优点也是无直流分量（全部转换为双极性电平，无零电平），最长连续“0”码或连续“1”码的个数仅为2，非常容易实现同步，编译码电路简单。因为如果原来基带信号是连续的“1”码，转换后都为“10”两个码了，不会出现连续的“1”码，如果原来基带信号是连续的“0”码，转换后都为“01”两个码了，不会出现连续的“1”码或“0”码。仅当原来基带信号是连续的“01”码时，才可能出现转换后的连续“0”码；仅当原来基带信号是连续的“10”码时，才可能出现转换后的连续“1”码，而且最多仅2个连续的“0”码或“1”码。

5.差分曼彻斯特码

当极性反转时曼彻斯特码会引起译码错误，为解决此问题，又引入了它的衍生码型——差分曼彻斯特码（**Differential Manchester Encoding**）。差分曼彻斯特编码是曼彻斯特编码的一种修改形式，也是每一码元的1/2码元位置必须进行极性跳变，其不同之处在于不同位置的码元转换方式有些不一样。

对于二进制信号中的第一个码元，与曼彻斯特码的转换方式一样，都是在1/2码元时进行电平极性跳变，即把0码转换成01，把1码转换成10。

后面的信号码就根据下面的规则进行跳变：

□如果本码为1，开始处的电平不跳变。也就是说如果上一个码元的后1/2个半码元是正电平，则从本码元开始处继续维持正电平，一直到1/2个码元时才跳变到负电平，即“1”码变为“10”码；如果上一个码元的后1/2个半码元是负电平，则从本码元开始处继续维持负电平，一直到1/2个码元时才跳变到正电平，即“1”码变为“01”码。

□如果本码为0，开始处的电平必须跳变。也就是如果上一个码元的后1/2个半码元是正电平，则从本码元开始处跳变为负电平，一直到1/2个码元时才跳变到正电平，即“0”码变为“01”码；如果上一个码元的

后1/2个半码元是负电平，则从本码元开始处跳变为正电平，一直到1/2个码元时才跳变到负电平，即“0”码变为“10”码。

从以上转换规则可以看出，除开始的一个信号码外，其他的信号码最终如何转换是不固定的，要根据对应码是“0”，还是“1”，以及它的前一个码转换后的后1/2电平极性而定。如原来基带数字信号代码为110001110，首先把开始信号码“1”转换为“10”，后面的码根据以上规则进行转换，最终结果如图4-31所示，从图中可以看出它与曼彻斯特码的区别。转换后的差分曼彻斯特码就为100101010110011010。

在差分曼彻斯特编码中，每一码元中间的跳变也只用来作为同步的时钟信号，所以它也是一种自同步编码。但它的时钟和数据是分离的，便于数据提取。差分曼彻斯特编码主要用于令牌环网。

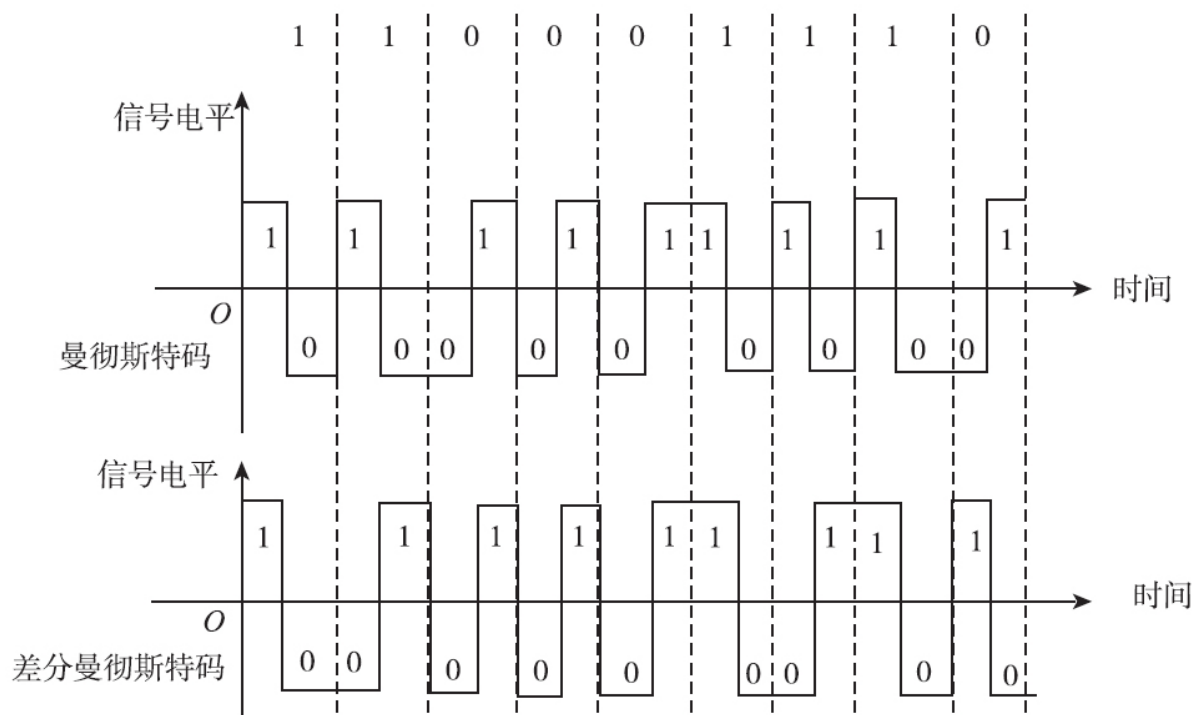


图 4-31 差分曼彻斯特码转换示例

4.5 信号调制与解调

在计算机网络中，大多数要传输的信号都是具有较低频带的信号，这种信号称为基带信号。如果用基带信号直接传输，则这种传输方式称为基带传输，这在本章前面已有介绍。但是，大多数信道不适宜进行基带信号的传输，因为这样的低频信号在传输中会产生很大的衰减和失真。在数字基带传输中，我们知道可以通过数字编码后在信道中直接传输数字信号，但这也仅适用于近距离传输（如在同一局域网中），在复杂网络环境中（如存在不同传输介质和信道类型的网络中），同样不能直接这样传输。因此，需要将基带信号进行调制，变换为适合信道传输的形式，调制是让基带信号去控制载波的某个（或某些）参数，使该参数按照信息的规律进行变化，这就是“调制”过程。“解调”是“调制”的逆过程，即从调制后的信号中恢复原来的调制信号的过程。

4.5.1 调制与解调的关键术语

在调制和解调过程中涉及几个非常关键的术语，下面先来介绍这几个术语。

1. 调制

首先我们要明白什么叫调制，为什么要进行调制。调制最通俗的解释就是用一种“能量大”的信号承载（可简单地理解为“背上”）另一种“能量小”的信号进行传播、传输。承载另一种信号的信号我们称为载波或者载波信号，而被承载信号才是我们真正要传播或者传输的信号。就像不会游泳的人坐在船上，由船渡我们过河一样，此时我们人就可以比做被承载信号，而船就可以比做载波信号。

但是，调制不是仅把被承载信号附加在载波信号上就万事大吉了，因为在传输系统中真正发挥作用，需要的还是被承载信号，载波信号只是一个载体，总得让真正有用的被承载信号发挥作用。这时就得通过被承载信号的有关特性来决定载波信号的幅度、频率或者相位。就像我们坐在船上不是由船自由漂流，而是由我们人来决定船的行驶方向、路径和速度等一样。当信号到了接收端时，我们又得把被承载信号释放出来（相当于我们人下船），恢复它的“自由身”，只有这样，接收端才能真正接收到有用的信号。这个释放的过程就是与调制相对的“解调”。

在对数字信号进行载波调制时，我们所采用的也是模拟载波调制中的调幅、调频和调相这三种调制技术。但它们对应已有了另外的名字：ASK（Amplitude Shift Keying，幅度键控）、FSK（Frequency Shift Keying，频率键控）和PSK（Phase Shift Keying，相位键控）。这里之所以都称之为“键控”，是指在这些调制技术中都是用电键进行

控制的，这是借用了电报传输中的术语。它们分别对应于利用载波（正弦波）的幅度、频率或相位来承载数字基带信号，可以看做是模拟线性调制和角度调制的特殊情况。

2.调制信号

调制信号就是用于对载波信号中某个参数进行特性控制，使其按照自己的对应参数特性变化的信号（也是最终要传输的有用信号），以便在接收端可以根据这个被控制的特性还原出原始的调制信号。

调制信号通常是低频信号，可以是模拟信号，也可以是数字信号，所以也就有对应的模拟信号调制（或模拟调制）和数字信号调制（或数字调制）。数字调制与模拟调制在本质上没有什么区别，都属于正弦波调制，只是源信号不同。数字调制中的源信号为离散型的脉冲数字信号，而模拟调制中的源信号为连续型的正弦波信号。在此仅介绍数字信号调制技术。

（1）载波信号

载波就是用来载送有用低频调制信号的信号波，通常是一种高频信号。通过调制技术就可以把调制信号和载波信号进行叠加，使载波信号的某些参数特性（如信号幅度、信号频率或信号相位等）按调制信号变化。具体是载波的哪个参数会随着调制信号发生变化，要视具体的调制类型而定，如果被控制的参数为幅度，则把这种调制称为调

幅；如果被控制的参数为频率，则把这种调制称为调频；如果被控制的参数为相位，则把这种调制称为调相。到达接收端后，通过解调技术从已调信号中分离出有用的数据即可。

那为什么要用载波进行调制？因为通常我们要发送的数字信号的频率是比较低的，如果按照本身的频率来传输，信号的衰减比较严重，不利于远距离传输。打个比方，如果我们要送一样物品（相当于这里所说的“调制信号”）到达某个比较远的地方，仅靠我们人力可能很困难，甚至根本无法达到目的（因为我们的速度很低，体力有限），但是如果我们使用像汽车、火车等这类交通工具来运输，就很轻松了，这时的交通工具就相当于我们这里所说的“载波”。使用载波传输，我们可以将数据的信号加载到载波的信号上，接收方按照载波的频率来接收数据信号。由于有用的数据源信号波的波幅与无用的载波信号波的波幅是不同的，通过进行波幅过滤就可以将数据源信号提取出。

（2）已调信号

已调信号是载波信号在被调制信号调制后所产生的新信号，它同时具备了原来调制信号和载波信号的双重特性（既不完全等同于原来的调制信号，也不完全等同于载波信号）。如在模拟信号调制中采用的是调幅这种调制技术，载波的振幅会受调制信号的控制，但频率、

相位仍是由载波信号决定的，所以最终的已调信号就是与调制信号振幅一致，频率和相位与载波信号一致的新信号。

(3) 解调

解调是调制的反过程，是从已调信号中通过某种技术（如低通滤波器等）恢复出原来调制信号的过程，因为我们最终有用的还是原来的调制信号。之所以要先经过调制过程，就是为了便于低频信号在信道中的远距离、高效传输。到了目的地（接收端）后，自然要把它还原出来。

4.5.2 ASK调制与解调

ASK（幅度键控）是一种数字幅度调制技术，是指正弦载波的幅度随数字基带信号变化而变化的数字调制，又称通断键控（on-offkeying, OOK）或者开关键控。当数字基带信号为二进制码时，称为二进制振幅键控（2ASK），但它仅适用于单极性数字信号。它是利用代表数字信息“0”或“1”的基带矩形脉冲去键控一个连续的载波，使载波时断时续地输出。即源数字基带信号为“1”时，发送载波信号，源数字基带信号为“0”时，发送零电平，也就是不发送载波信号。

1.2ASK调制原理

2ASK调制就是以不同的波幅来表示源信号中的不同二进位值，相当于用一个如图4-32所示的开关电路来控制载波信号的输出。这个开关就是二进制基带调制信号中的“0”和“1”，当调制信号为“1”码时，相当于开关接通电路，有载波输出；当调制信号为“0”时，相当于开关断开电路，无载波输出。上述是2ASK的键控法，另外还有一种模拟法，采用图4-33所示的乘法器把数字基带信号与正弦载波进行乘法运算得出。 $\cos\omega_C t$ 为正弦载波信号， $s(t)$ 为二进制单极性不归零码。

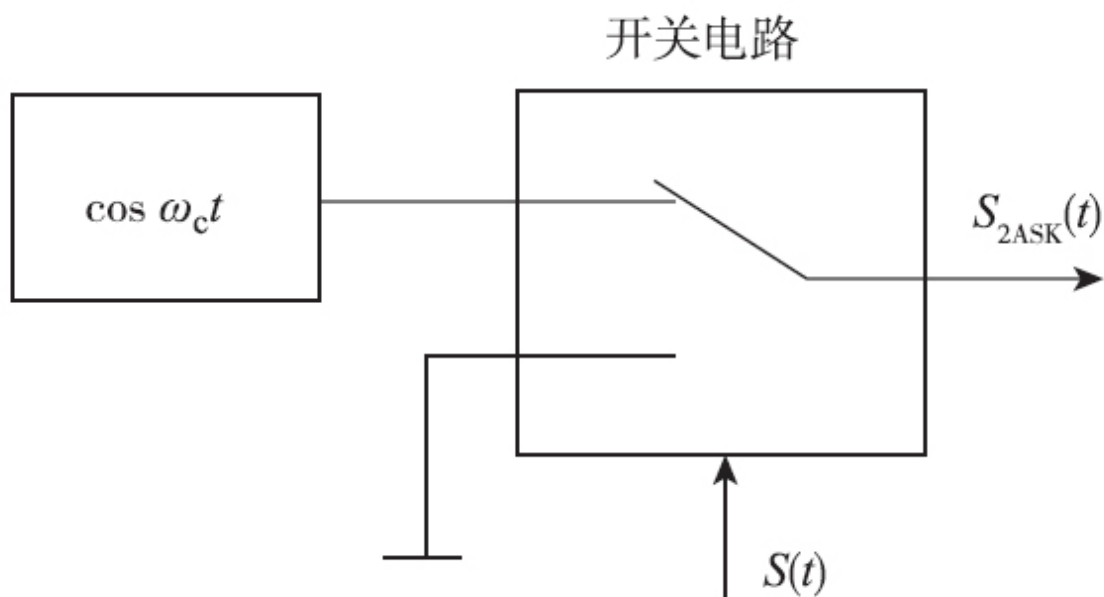


图 4-32 2ASK键控法调制模型

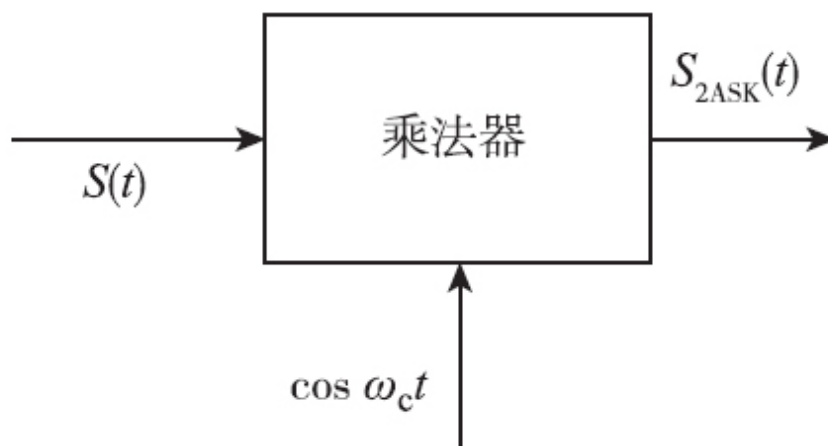


图 4-33 2ASK模拟法调制模型

图4-34所示的就是一个2ASK调制示例，示例中二进制基带调制信号为1011001，载波为一正弦波。根据以上2ASK的调制原理，很快就可以得到最终的已调制信号波形。

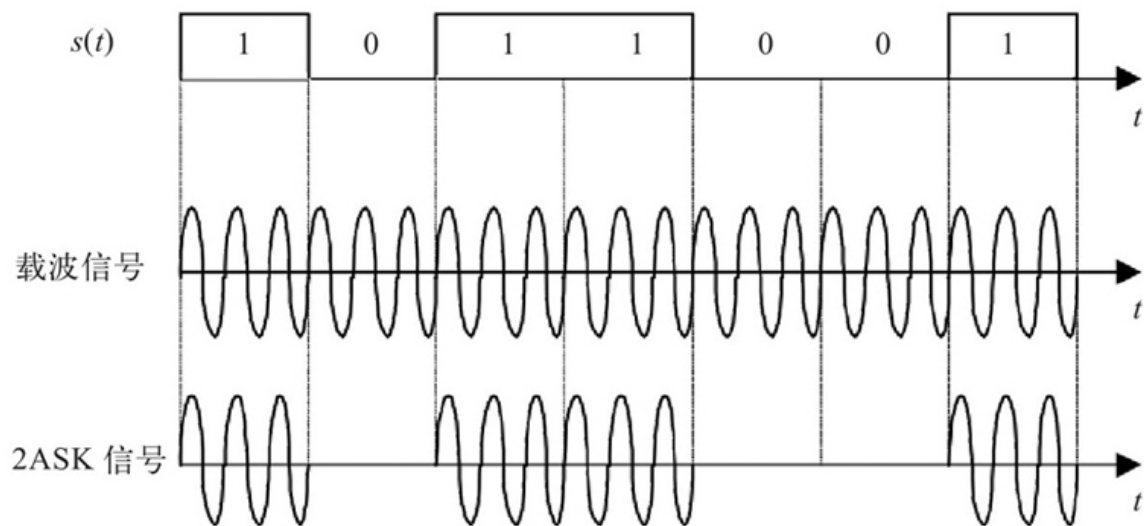


图 4-34 2ASK调制示例

2ASK信号是数字调制方式中最早出现的，也是最简单的，但其抗噪声性能较差（因为它仅是根据调制信号来控制输出信号的幅度，并不能过滤干扰信号），因此实际应用并不广泛，但经常作为研究其他数字调制方式的基础。

2.2ASK解调

与2ASK调制有键控法和模拟法两种方式类似，2ASK信号解调也有两种方式：非相干解调（包络检波法）和相干解调（同步检测法）。相干解调是指要利用乘法器，输入一路与载波相干（同频同相）的参考信号，然后再与载波相乘，最终得到解调信号，这种方法常用于线性调制信号，如ASK和PSK；所谓非相干解调即不需提取载波

信息（或不需恢复出相干载波）的一种解调方法，这种方法主要用于FSK，也可用于ASK。

(1) 非相干解调

2ASK的非相干解调模型如图4-35所示，与模拟信号解调模型的不同点仅在于多了一个采样判决器，因为这是将已调制的连续信号（参见图4-34）还原为原始的离散二进制数字信号，所以需要对原来已调制信号进行采样。图4-34中的 $y_i(t)$ 表示的是已调信号。图4-34中为各步所生成的信号进行了标注，以便于波形说明。

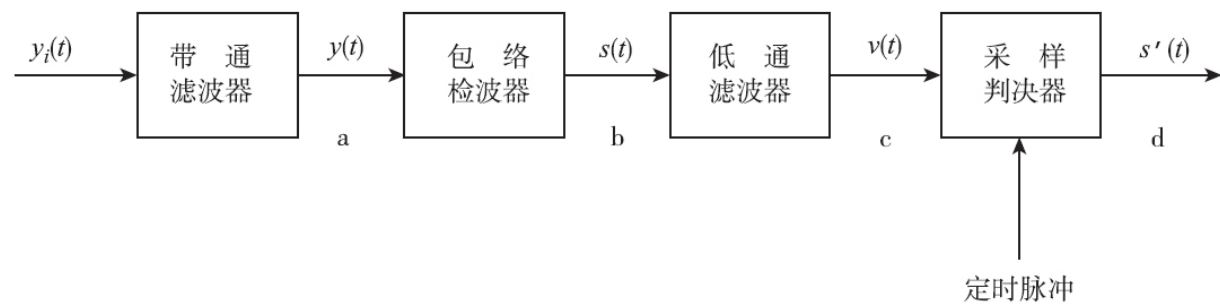


图 4-35 2ASK非相干解调模型

图4-36所示的是将一个二进制单极性调制信号（源信号）110010001010调制后并解调还原出调制信号的过程。具体步骤如下（同时参照图4-35）：

1) 已调信号 $y_i(t)$ 经过带通滤波器（主要用于过滤掉已调信号的低频和低频干扰信号）后得到图4-36中的a波；

2) a波再经过包络检波器进行包络检测（“包络”就是外部表现的意思，这里的“包络检测”就是检测出信号的幅度变化的曲线，不管频率和相位特性）后得到了b波。对比b和a波可以看出，它是一个极性过滤，因为调制信号是单极性的，所以这里仅允许正极性的波形输出；

3) b波再经过低通滤波器后就得到了c波。这里的低通滤波器用于过滤b波中高频成分，仅允许符合源信号频率特性的波形输出；

4) 最后经过采样判决器，对连续型的c波进行采样，还原出原始的离散型调制信号d。

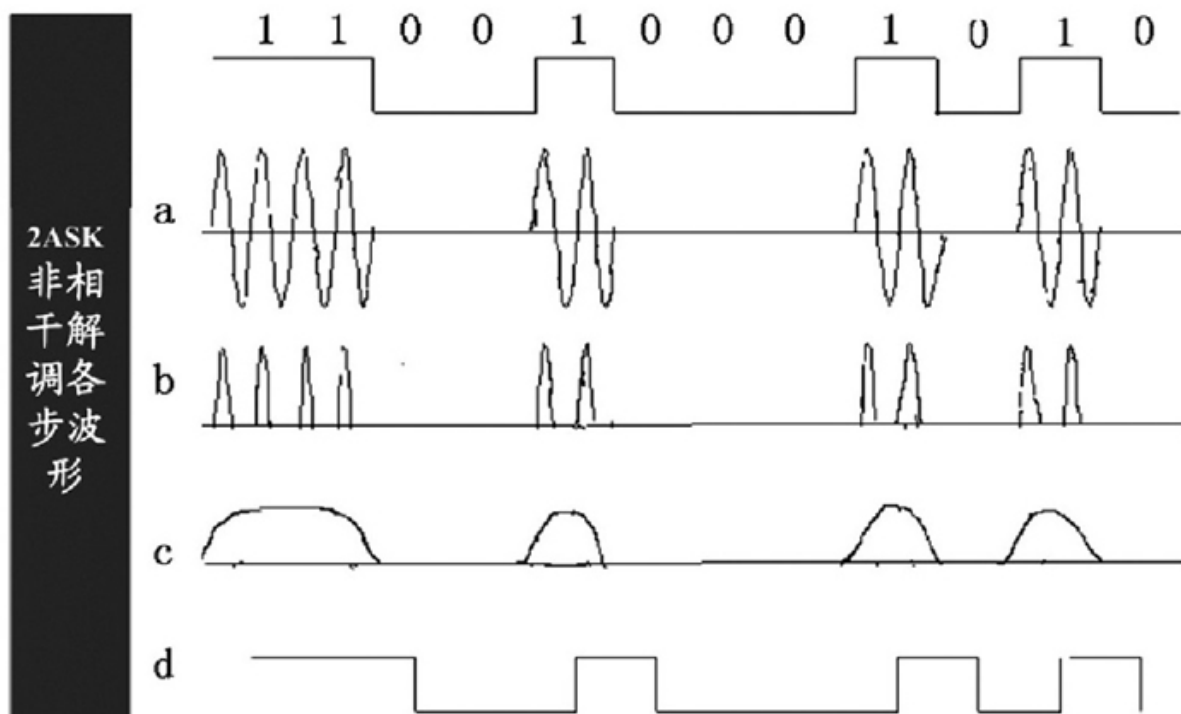


图 4-36 2ASK非相干解调示例

从以上解调过程可以看出，2ASK非相干解调没有利用载波信号来影响波形的输出。下面介绍2ASK的相干解调。

(2) 相干解调

2ASK的相干解调模型如图4-37所示，与模拟信号解调模型的不同点也仅在于多了一个采样判决器，原因同上。图4-37中的 $y_i(t)$ 表示的也是已调信号。图4-37中同样为各步所生成的信号进行了标注。

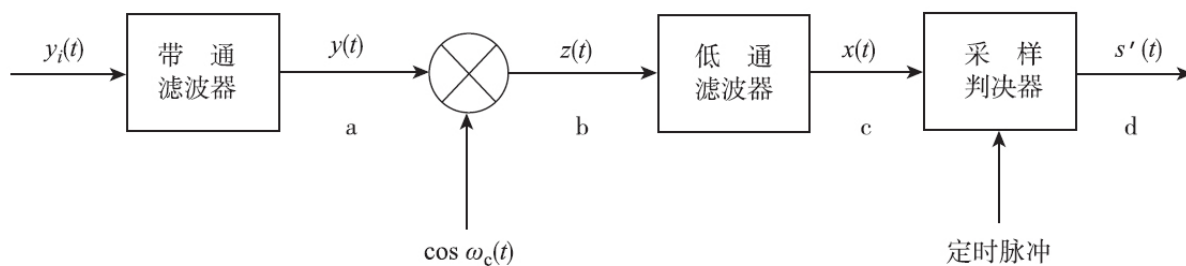


图 4-37 2ASK相干解调模型

图4-38所示的也是前面介绍的一个二进制单极性调制信号（源信号）110010001010，解调后还原出调制信号的过程。具体步骤如下（同时参照图4-37）：

1) 已调信号 $y_i(t)$ 经过带通滤波器（主要用于过滤掉已调信号的低频和高频干扰信号）后得到图中的a波。这与非相干解调步骤是一样的。

2) a波再与载波信号 $\cos\omega_c t$ 在乘法器中进行相乘得到b波。对比b和a波可以看出，它不仅是一个极性过滤，而且包含了载波的频率成分，

信号更强；

3) b波再经过低通滤波器后就得到了c波。这里的低通滤波器用于过滤b波中高频成分，仅允许符合源信号频率特性的波形输出，这一步与“非相干解调”也一样。

4) 最后经过采样判决器，对连续型的c波进行采样，还原出原始的离散型调制信号d，这步也与“非相干解调”一样。

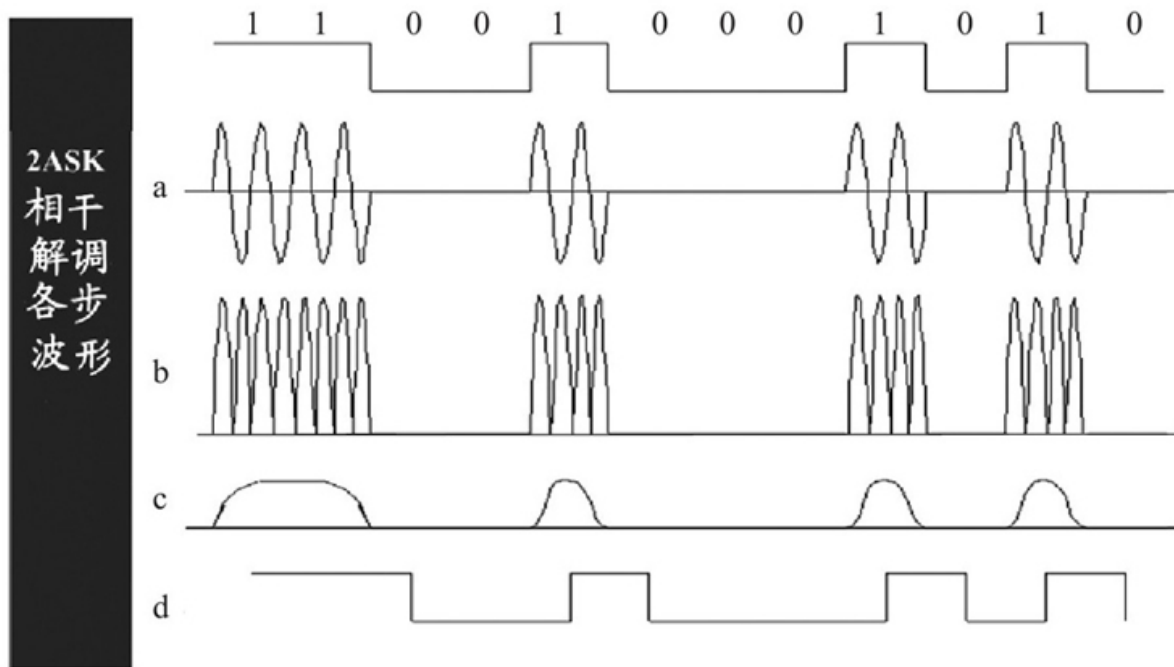


图 4-38 2ASK相干解调示例

从以上叙述可以看出，非相干解调和相干解调在执行步骤上唯一的区别就是第2步，相干解调借助了载波信号，主要是为了使信号得到加强，更便于在后面的带通滤波器中过滤。

4.5.3 FSK调制与解调

FSK（频率键控）是用数字基带信号去控制载波的频率特性来传送信息的一种调制方式，但它仅适用于双极性数字信号。它把信号的振幅、相位作为常量，而把频率作为变量，通过频率的变化来实现信号的识别。如数字信号的“1”码用频率为 f_1 的载波（载波1）来传送，“0”码用频率为 f_2 的载波（载波2）来传送，相当于载波在两种不同频率之间进行切换，故又称频移键控。

1.2FSK调制

在二进制数字调制中，若正弦载波的频率随二进制基带信号在两个频率点间变化，则产生的是二进制移频键控信号（2FSK信号）。2FSK信号的产生与2ASK信号的产生一样，从原理上讲，数字调频可用模拟调频法来实现，也可用键控法来实现。既可以采用模拟调频电路来实现（称为2FSK模拟法调制或者直接调频法，是利用一个矩形脉冲对载波进行频率调制），又可以采用数字键控的方法来实现（称为2FSK键控法调制或者移频键控法，是利用受控的矩形脉冲序列控制的开关电路对两个独立的频率源进行选通），分别如图4-39和图4-40所示。 $S(t)$ 为调制信号， $S_{2FSK}(t)$ 为已调的2FSK信号。



图 4-39 2FSK模拟法调制模型

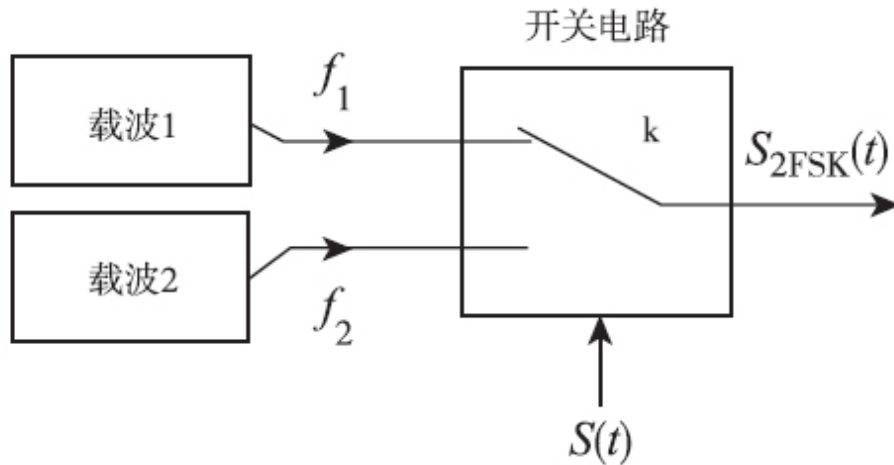


图 4-40 2FSK键控法调制模型

所谓“直接调频法”，就是用数字基带矩形脉冲控制一个振荡器的某些参数，直接改变振荡频率，使输出得到不同频率的已调信号。用此方法产生的2FSK信号对应着两个频率的载波，在码元转换时刻，两个载波相位能够保持连续，所以称其为相位连续的CPFSK

（Continuous-Phase Frequency Shift Keying，连续相位频移键控）信号。因为只用到了一个载波，所以调制后的 $S_{2FSK}(t)$ 信号码元间的相位是连续的，如图4-41所示。

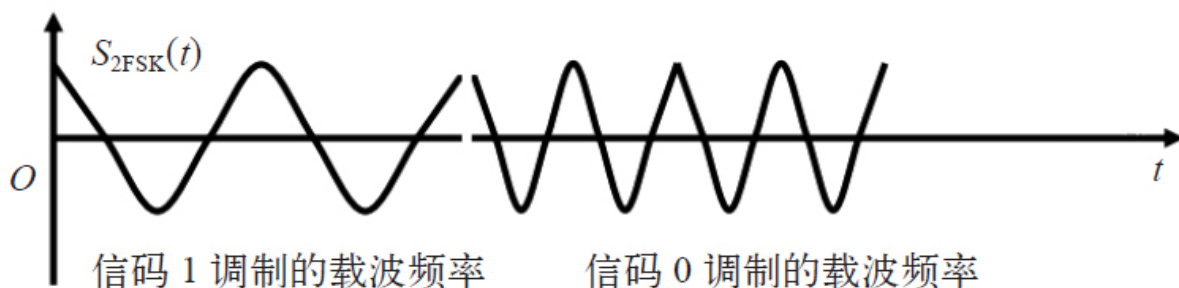


图 4-41 相位连续的 $S_{2FSK}(t)$ 信号

频率键控法又称为频率转换法，是用数字矩形脉冲控制电子开关，使电子开关在两个独立的振荡器（产生两种不同频率的载波 f_1 和 f_2 ）之间进行转换，从而在输出端得到不同频率的已调信号。由于产生 f_1 和 f_2 载频是由两个独立的振荡器实现的，所以码元间的相位不一定是连续的，如图4-42所示。这种方法的特点是转换速度快、波形好、频率稳定度高、电路不甚复杂，在实用中可以用一个频率合成器代替两个独立的振荡器，再经分频链进行不同的分频，也可得到2FSK信号。

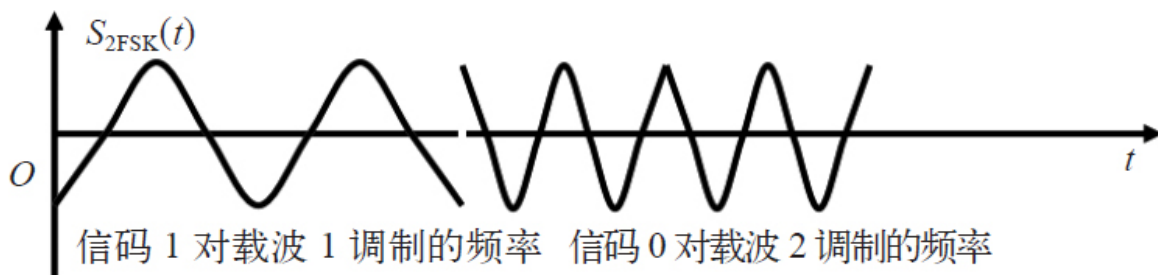


图 4-42 相位不连续的 $S_{2FSK}(t)$ 信号

2FSK信号波可看做两个2ASK信号波的合成，信码1和0分别调制不同频率的载波。但因为2FSK调制的是双极性数字信号，所以我们可以

把一个单极性2ASK调制信号分成两个互反的信号（否则原来单极性信号中的0码无法对载波进行调制）。

图4-43所示就是源信号码字为1011001的两个互反的信号 $s(t)$ 和 $\overline{s(t)}$ 波形。然后用这两路信号作为调制信号，分别对两个不同频率的载波进行调制，如图4-44所示。其中 $c_1(t)$ 和 $c_2(t)$ 分别代表两个不同的载波， $s_1(t)$ 和 $s_0(t)$ 是 $s(t)$ 和 $\overline{s(t)}$ 这两路调制信号分别对 $c_1(t)$ 和 $c_2(t)$ 载波进行2ASK调制后的波；再把 $s_1(t)$ 和 $s_0(t)$ 这两路信号进行叠加就得到了最终的已调信号2FSK。

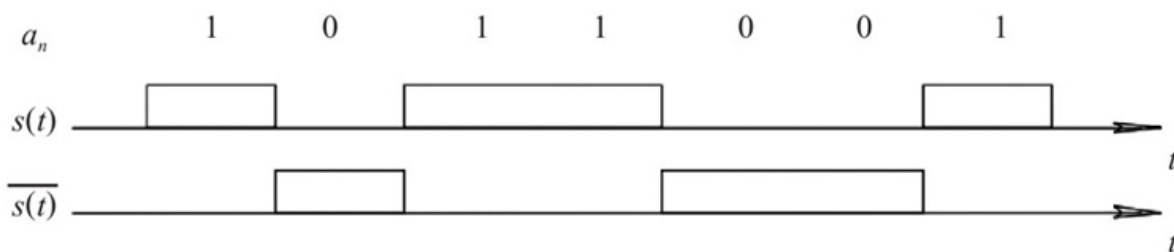


图 4-43 互反的两路单极性数字基带信号

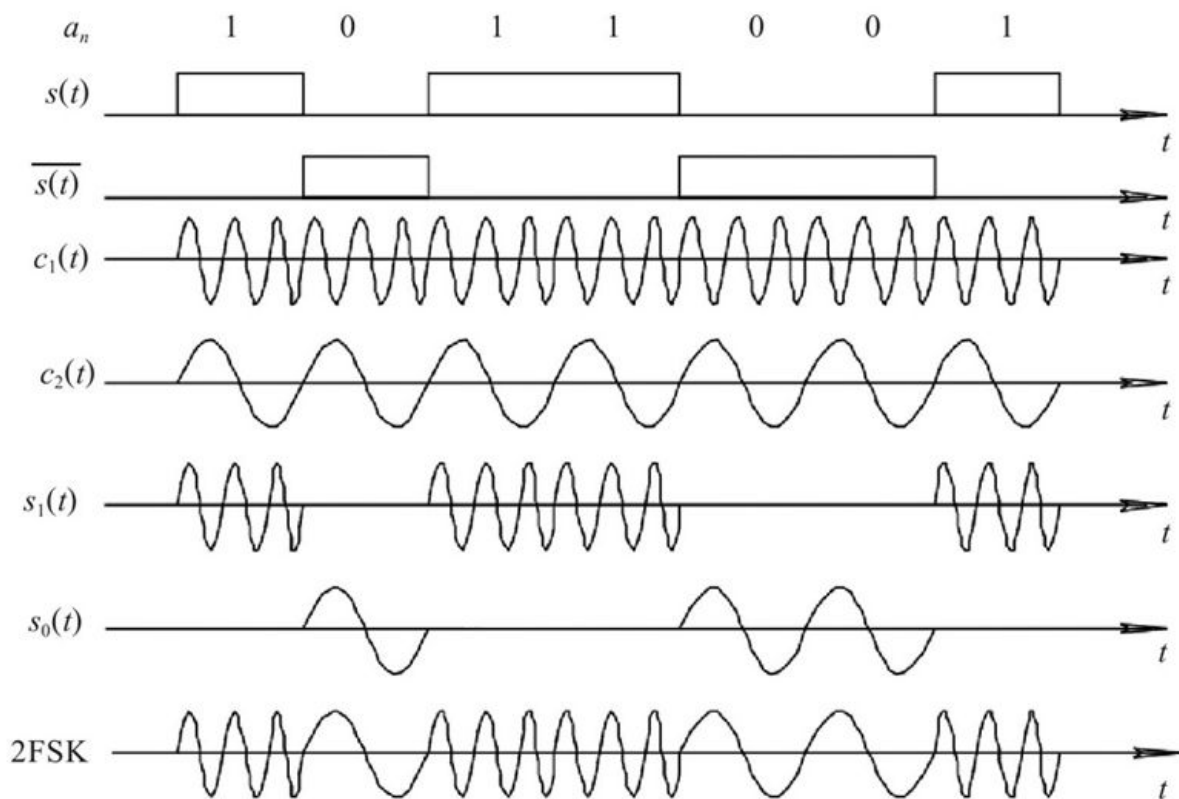


图 4-44 2FSK调制示例

2.2FSK解调

2FSK信号的解调方法很多，如鉴频法、过零检测法、非相干解调法（又称滤波检测法）和相干解调法（又称差分检波法）。这里仅介绍相干解调、非相干解调和过零检测这三种2FSK解调方法。

(1) 相干和非相干2FSK解调

2FSK信号的非相干解调（也称包络检测法）模型如图4-45a所示，其可视为由两路2ASK解调电路组成（比较图4-35）。这里的两个带通

滤波器（带宽相同，皆为相应的2ASK信号带宽；但中心频率不同，分别为 f_1 和 f_2 ）起分路作用，用以分开两路2ASK信号，经包络检测（这里的“包络检测”就是检测出信号的频率变化，不管幅度和相位特性）后分别取出互反的 $s(t)$ 及 $\overline{s(t)}$ 两路信号；采样判决器起比较器的作用，通过把两路包络信号同时送到采样判决器进行比较，就可以判决输出基带数字信号。现把上支路 $s(t)$ 和下支路 $\overline{s(t)}$ 的采样值分别用 V_1 、 V_2 表示，若采样判决器的判决准则为 $V_1 \geq V_2$ ，则输出信码1； $V_1 < V_2$ ，则输出信码0。

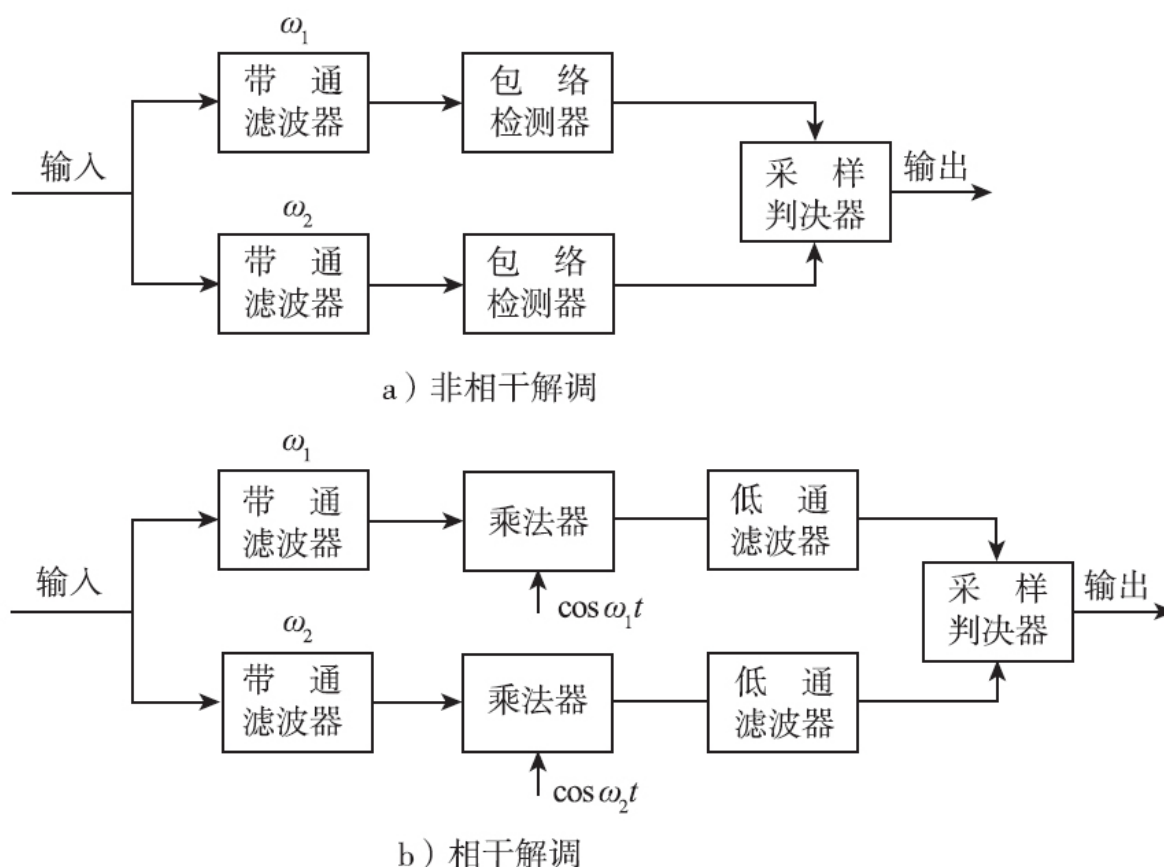


图 4-45 相干和非相干2FSK解调模型

图4-46所示的是一个基带数字信号11001000101经过2FSK调制后的2FSK信号通过非相干解调方法还原出数字基带信号的波形变化图。其中的 f_1 路检波和 f_2 路检波分别代表两路2FSK信号在经过两个频率不同的带通滤波器后输出的波形； $s(t)$ 和 $\overline{s(t)}$ 分别代表从两个带通滤波器输出的信号再经过包络检波器后输出的波形；最后在采样判决器上进行采样（两路信号的采样电压值分别用 V_1 和 V_2 表示）和判决，若 $V_1 \geq V_2$ 则输出信码1，若 $V_1 < V_2$ 则输出信码0。这样就可以还原出原始的数字基带信号了。

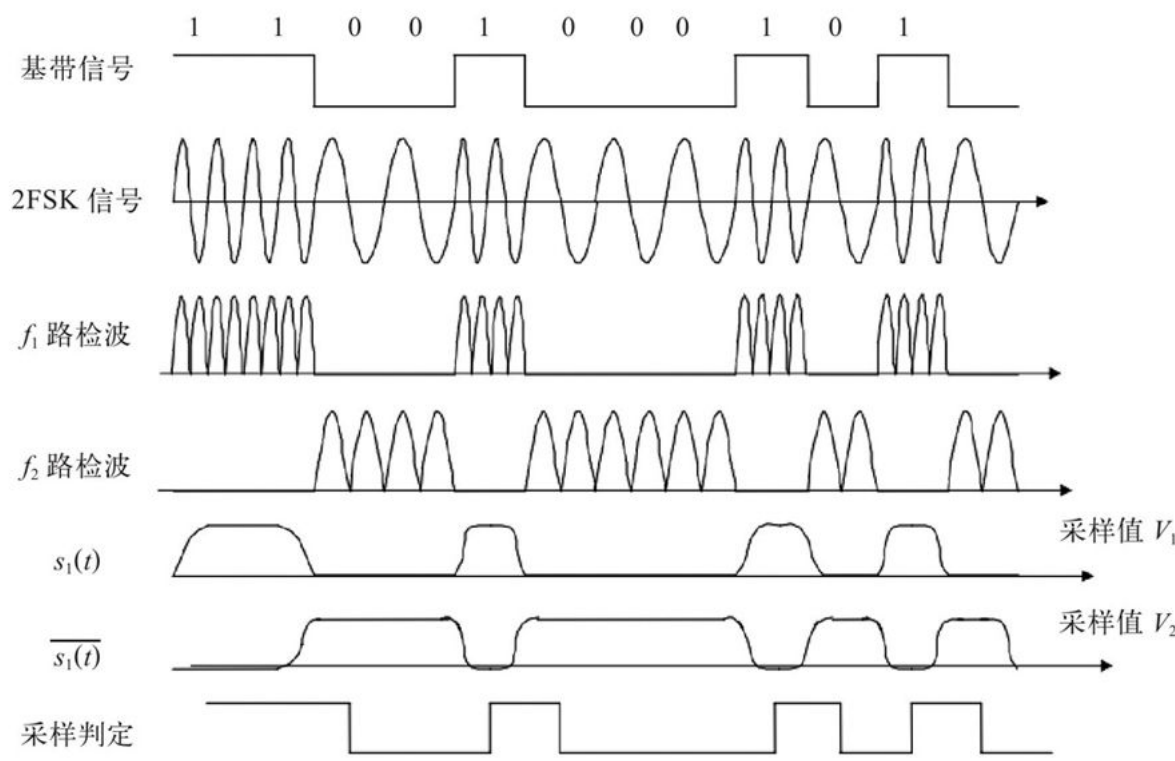


图 4-46 非相干2FSK解调示例

相干解调2FSK的模型如图4-45b所示。图中的两个带通滤波器起分路作用。它们的输出分别与相应的同步相干载波相乘，再分别经低通滤波器滤掉二倍频信号，取出含基带数字信息的低频信号，采样判决器在采样脉冲到来时对两个低频信号的采样值进行比较判决（判决规则与上面介绍的非相干解调）即可还原出基带数字信号。

(2) 过零检测法

单位时间内信号经过零点的次数多少，可以用来衡量频率的高低。数字调频波的过零点数随不同载频而异，故检出过零点数可以得到关于频率的差异，这就是过零检测法的基本思想。过零检测法2FSK解调模型如图4-47所示。各步的输出波形如图4-48所示。下面具体解释这些步骤，但是具体的实现方法在此我们不做研究。

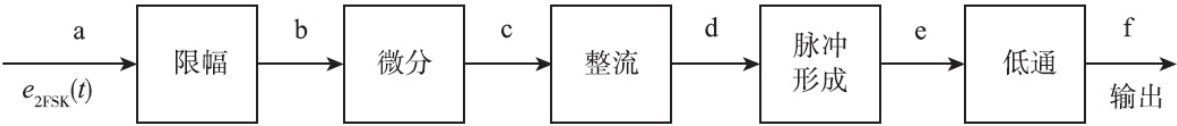


图 4-47 过零检测法2FSK解调模型

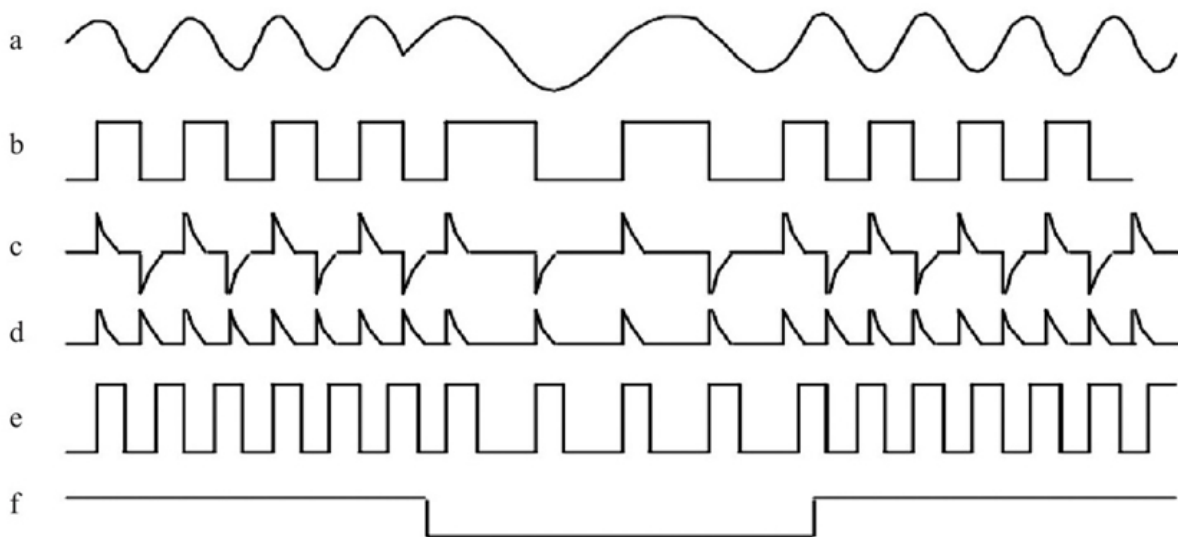


图 4-48 2FSK过零检测法解调示例

- 1) 2FSK输入信号a经放大、限幅后产生矩形脉冲信号b。
- 2) b信号经微分及整流后形成与频率变化相应的尖脉冲序列（分别对应信号c和d），这个序列就代表着调频波的过零点。
- 3) d尖脉冲再触发一脉冲发生器，变换成具有一定宽度的矩形波e。该矩形波的直流分量便代表着信号的频率，脉冲越密，直流分量越大，反映着输入信号的频率越高。
- 4) e矩形波经低通滤波器就可得到脉冲波的直流分量f。这样就完成了频率-幅度转换，再根据直流分量幅度上的区别还原出数字信号“1”和“0”。

4.5.4 PSK调制与解调

PSK（相位键控）是利用双极性数字基带信号对载波相位进行控制来传送信息的一种调制方式，也就是用载波的不同相位来表示信号的信码。我们知道，如果两个频率相同的载波同时开始振荡，这两个频率同时达到正最大值，同时达到零值，同时达到负最大值，此时它们就处于“同相”状态；如果一个达到正最大值时，另一个达到负最大值，则称为“反相”。一般把信号振荡一次（一周）作为 360° 。如果一个波比另一个波相差半个周期，我们说两个波的相位差 180° ，也就是反相。因此数字调相就是让载波在两种相位间切换，故又称相移键控。

如果调制信号是二进制数字基带信号，则这种**PSK**称为**2PSK**。**PSK**分为绝对相位键控（**Absolute Phase Shift Keying, APSK**）和相对相位键控（或者“差分相位键控”，**Differential Phase Keying, DPSK**）两种。因为通常采用的就是绝对相位键控方式，所以**APSK**通常表示为**PSK**。

1.绝对相位调制与解调

绝对相位键控（通常就表示为**PSK**，对应的二进制绝对相位键控就为**2PSK**）是用未调载波的相位为基准相位的**PSK**，此时在接收系统中必须有一个与发送系统相同的基准相位作为参考，以识别接收到的

是“1”码还是“0”码。调制规则是：当已调载波相位与基准相位相差 0° 时发送信码“1”，当已调载波相位与基准相位相差 180° 时发送信码“0”。

(1) 2PSK调制

2PSK信号的产生也有模拟调相法和键控调相法两种，模拟调相法是把2PSK看做双极性不归零码基带信号的数字调幅2ASK，即基带信号与载波的乘积，其调制模型如图4-49所示；而键控调相法是通过载波在两种不同相位之间进行切换而进行的，其调制模型如图4-50所示。其中 $s(t)$ 为数字基带信号， $\cos\omega_c(t)$ 为载波信号， $S_{2PSK}(t)$ 为已调2PSK信号。

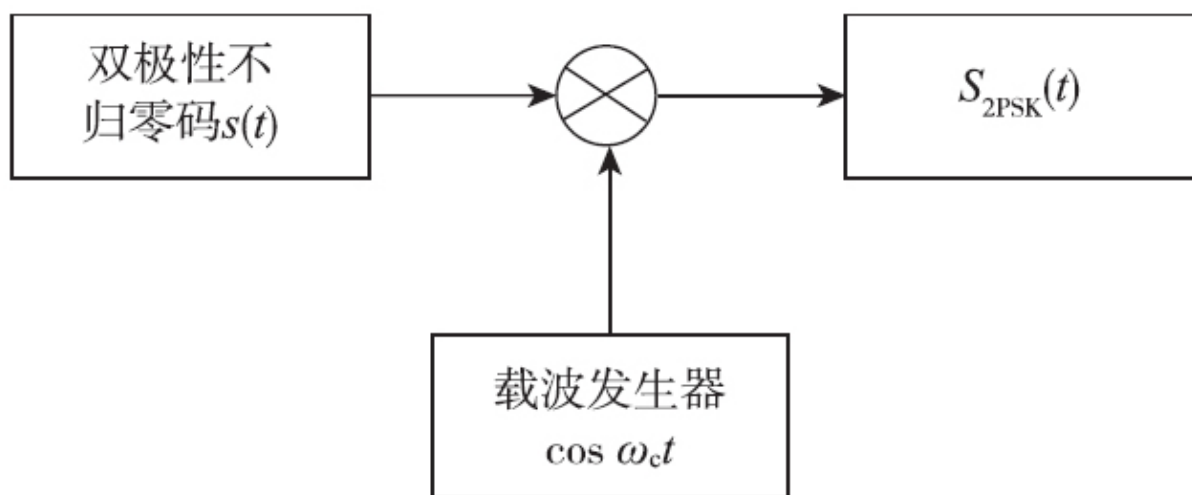


图 4-49 模拟调相法调制模型

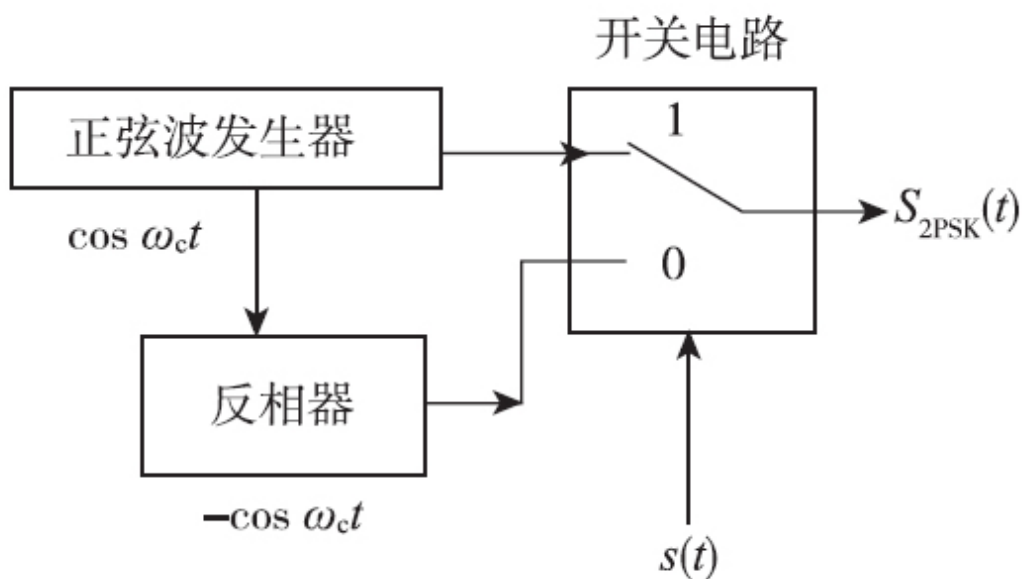


图 4-50 键控调相法调制模型

图4-51所示是一个二进制数字基带信号1011001的2PSK的调制示例，其中 $s(t)$ 就是数字基带信号的矩形脉冲波形， $c(t)$ 是一个正统载波波形，2PSK是经过相位调制后的已调波形。从图4-50中可以看出，当信码为1时，2PSK波形的相位与载波一样，而当信码为0时，2PSK的相位与载波相位相反。这种调制原理很好理解。

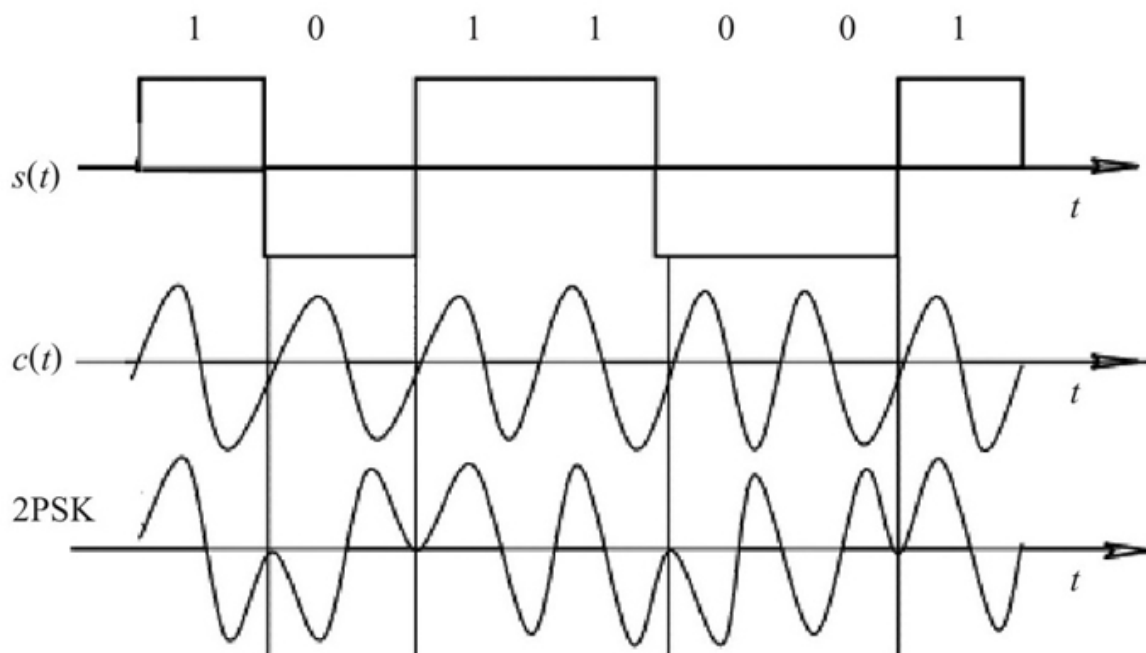


图 4-51 2PSK调制示例

(2) 2PSK解调

2PSK信号的解调通常都是采用“相干解调法”，其解调模型如图4-52所示。在解调过程中需要用到与接收的2PSK信号同频、同相的相干载波。如图4-53所示的是一个2PSK相干解调示例。

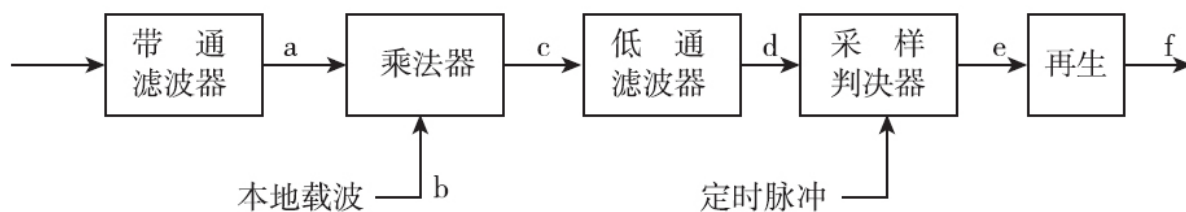


图 4-52 2PSK相干解调模型

在图4-53所示的示例中，a是2PSK已调信号经过带通滤波器后得到的信号波（带通滤波器主要用来过滤低频和高频干扰成分）；b是用来与2PSK信号相乘的本地相干载波；c是a与b经过相乘后得到的信号波；d是c信号波经过低通滤波器（去除高频成分）后得到的信号波；e是d信号经过数字采样后得到的离散矩形脉冲；f是经过再生器放大后得到的数字脉冲信号，这就是原始的基带数字信号。

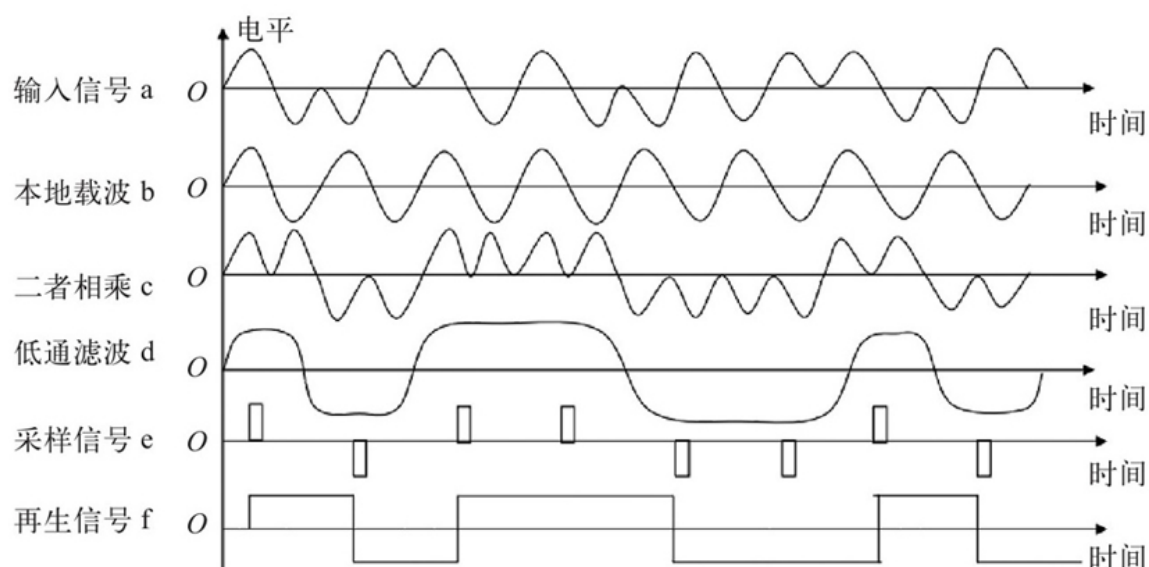


图 4-53 2PSK相干解调示例

相干解调需要一个与发送端调制时所用的载波同频同相的本地载波。但此载波是由接收端的载波提取电路提取。这里出现了一个问题，如果两端载波存在频率和相位偏差，就可能出现解调错误，如原来所有的“1”码都变成了“0”码，所有的“0”码都变成了“1”码。如图4-54所示的左、上图分别是采用与发送端载波相位相同（相差为0）、相反

（相差为 180° ）的本地载波2PSK调制结果，从中可以看出，它们的结果（f波形）是完全相反的。

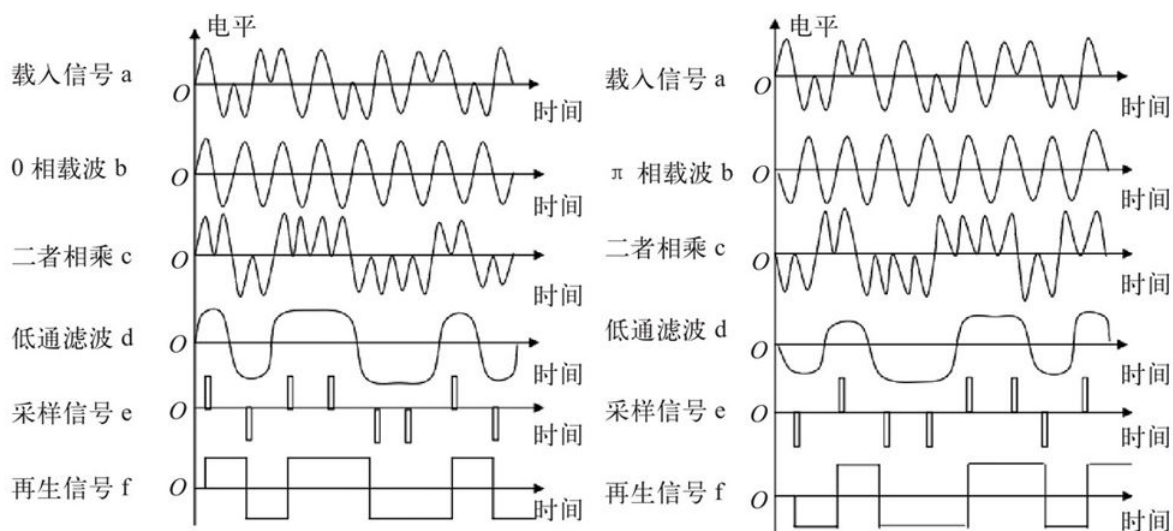


图 4-54 采用 0° 相差和 180° 相差本地载波解调的结果比较

2.相对相位调制与解调

为了解决绝对相位调制中的相位倒置问题，在进行数字调相之前先进行差分编码，再对差分码进行二元数字调相，这就是这里要介绍的二元差分相位调制（2DPSK）。

2DPSK不是利用载波相位的绝对数值传送数字信息，而是用前后码元的相对载波相位值来传送数字信息。所谓相对载波相位是指本码元初相位与前一码元初相位之差，即相位偏移，并规定：当相位偏移值为 0 时输出信码 0 ，相位偏移 180° 时输出信码 1 。

(1) 2DPSK调制

2DPSK信号的产生过程：首先对数字基带信号进行差分编码，即由绝对码变为相对码（差分码），然后再进行绝对调相。所以，2DPSK的调制原理与2PSK的调制原理类似，也是用二进制基带信号作为模拟开关的控制信号轮流选通不同相位的载波来完成2DPSK调制，不同的仅是先要对数字基带信号进行差分编码。其调制模型如图4-55所示，与如图4-50所示的绝对调相模型非常相似。

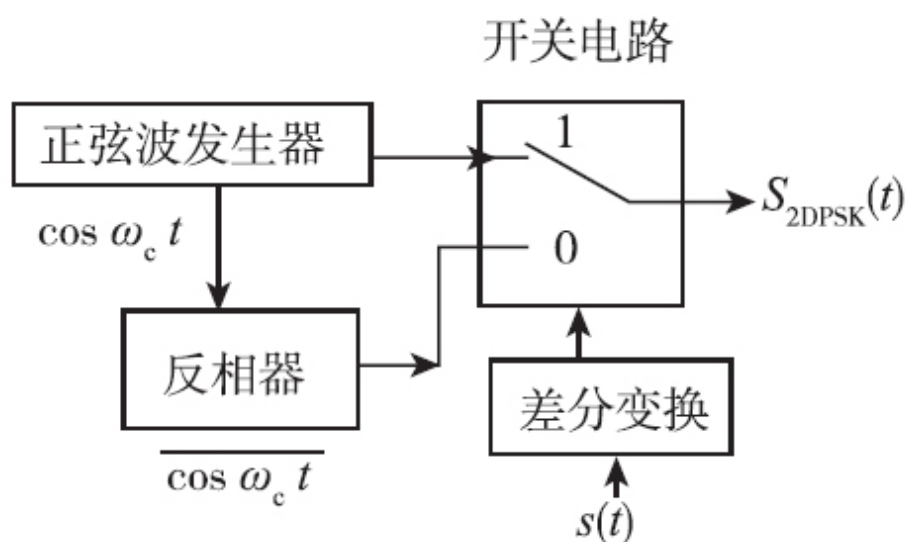


图 4-55 2DPSK调制模型

图4-56所示的是一个2DPSK调制示例。图中首先是输入的数字基带信号（绝对码），然后通过差分变换后得到“相对码”信号波。这里的“相对码”有一个计算规则，即模2逻辑加法运算：

$$b_n = a_n \oplus b_{n-1}$$

式中 b_n 为当前码的相对码， a_n 为当前码的绝对码， b_{n-1} 是 b_n 的前一个码元的相对码，最初的 b_{n-1} 可任意设定。模2逻辑加运算规则是： $0+0=0$ ， $0+1=1$ ， $1+0=1$ ， $1+1=0$ ，也就是只要双方值不一样时结果才是1，相同则结果为0。

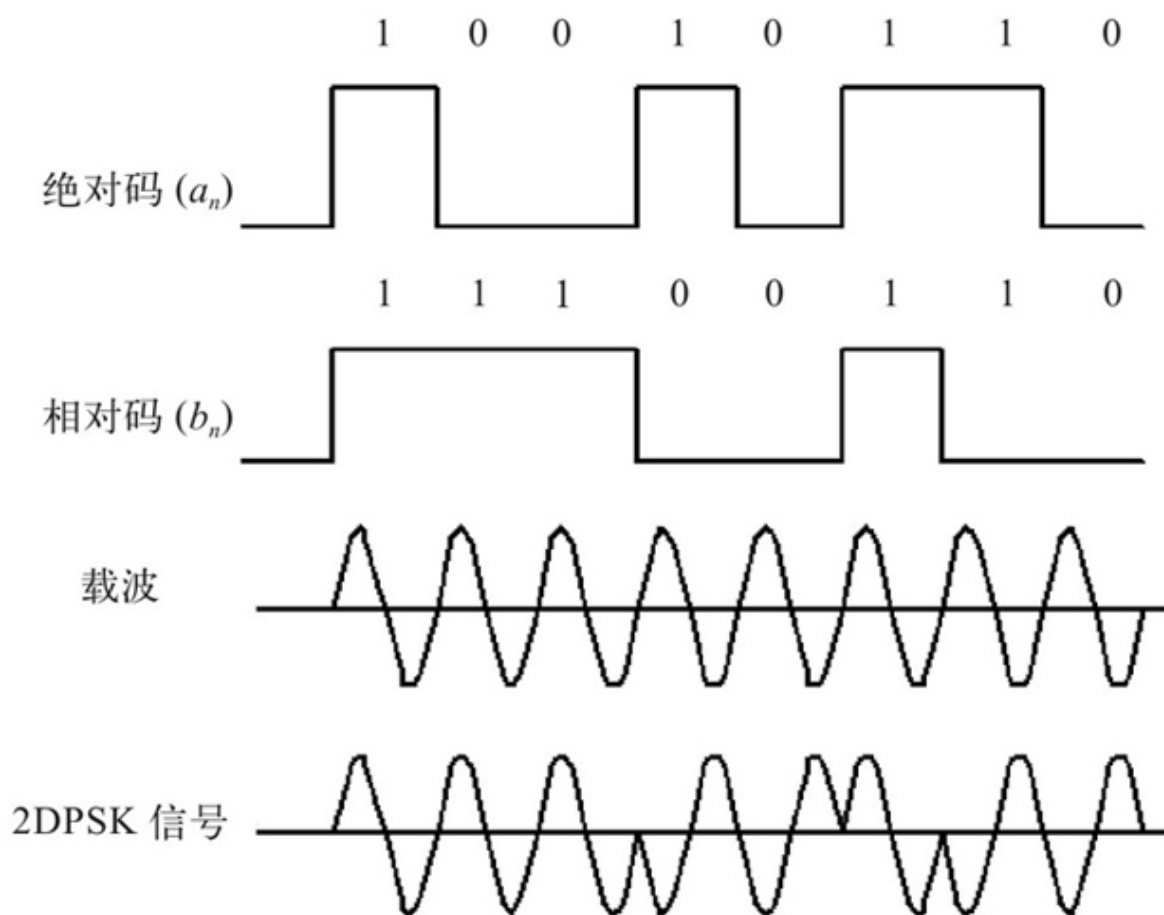


图 4-56 2DPSK调制示例

下面来分析一下图4-56所示的示例，假设最初的 b_{n-1} 相对码为0。

第一个码元的绝对码 (a_1) 为1, 因为最初的 b_0 相对码为0, 所以它们模2加运算的结果为1, 即 $b_1=1$ 。

第二个码的绝对码 (a_2) 为0, 因为 b_1 相对码前面已算出为1, 所以它们模2加运算的结果也为1, 即 $b_2=1$ 。

第三个码的绝对码 (a_3) 为0, 因为 b_2 相对码前面已算出为1, 所以它们模2加运算的结果同样为1, 即 $b_3=1$ 。

第四个码的绝对码 (a_4) 为1, 因为 b_3 相对码前面已算出为1, 所以它们模2加运算的结果为0, 即 $b_4=0$ 。

第五个码的绝对码 (a_5) 为0, 因为 b_4 相对码前面已算出为0, 所以它们模2加运算的结果为0, 即 $b_5=0$ 。

第六个码的绝对码 (a_6) 为1, 因为 b_5 相对码前面已算出为0, 所以它们模2加运算的结果为1, 即 $b_6=1$ 。

第七个码的绝对码 (a_7) 为1, 因为 b_6 相对码前面已算出为1, 所以它们模2加运算的结果为0, 即 $b_7=0$ 。

第八个码的绝对码 (a_8) 为0, 因为 b_7 相对码前面已算出为0, 所以它们模2加运算的结果为0, 即 $b_8=0$ 。

这样最终就得出数字基带信号10010110的相对码为11100100。

得到了相对码，后面载波的变化（也就是2DPSK信号的相位）就很容易了，因为它也有一个与相对码对应的计算规则，那就是：当相对码“1”时，调制信号的相位与原载波一样，当相对码为“0”时反转 180° 。这样就很容易画出图4-56所示最后的2DPSK波形。

(2) 2DPSK解调

2DPSK解调最常用的方法就是极性比较法（相干解调）和相位比较法（差分解调）两种，其解调模型分别如图4-57和图4-58所示。

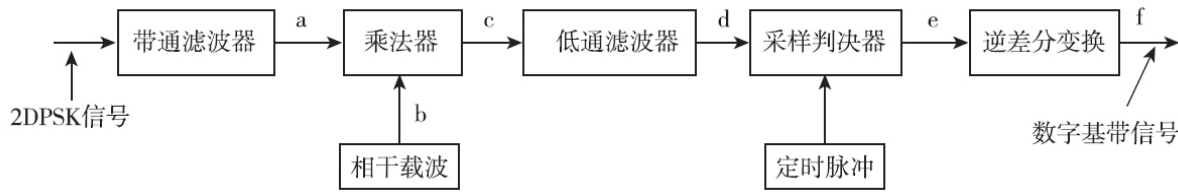


图 4-57 2DPSK相干解调模型

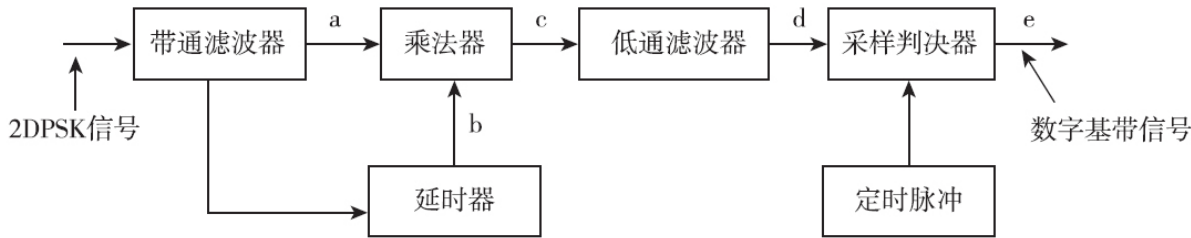


图 4-58 2DPSK差分解调模型

2DPSK解调是2DPSK调制的逆过程。前面我们说了，2DPSK调制过程中要对绝对码（也就是“数字基带信号”）进行差分处理，将其转换成相对码，然后再进行绝对相位调制。2DPSK解调则需要先把2DPSK已调信号转换成相对码，然后再把相对码转换成绝对码，这个

绝对码就是要解调出来的数字基带信号。下面仅对相干解调法进行介绍。

图4-59所示是一个2DPSK相干解调示例。图中a为2DPSK已调信号经过带通滤波器（把高、低频干扰成分过滤掉）得到的2DPSK信号。它经过与本地相干载波相乘后得到c信号，然后再经过低通滤波器过滤高频成分得到d信号；d信号经过定时采样后得到离散的二进制脉冲信号e（这就相当于抽取到了调制信号中的“相对码”，本示例中为111001001），然后再逆差分变换就得到了原始的数字基带信号（也就是绝对码，本示例中为000101100）。

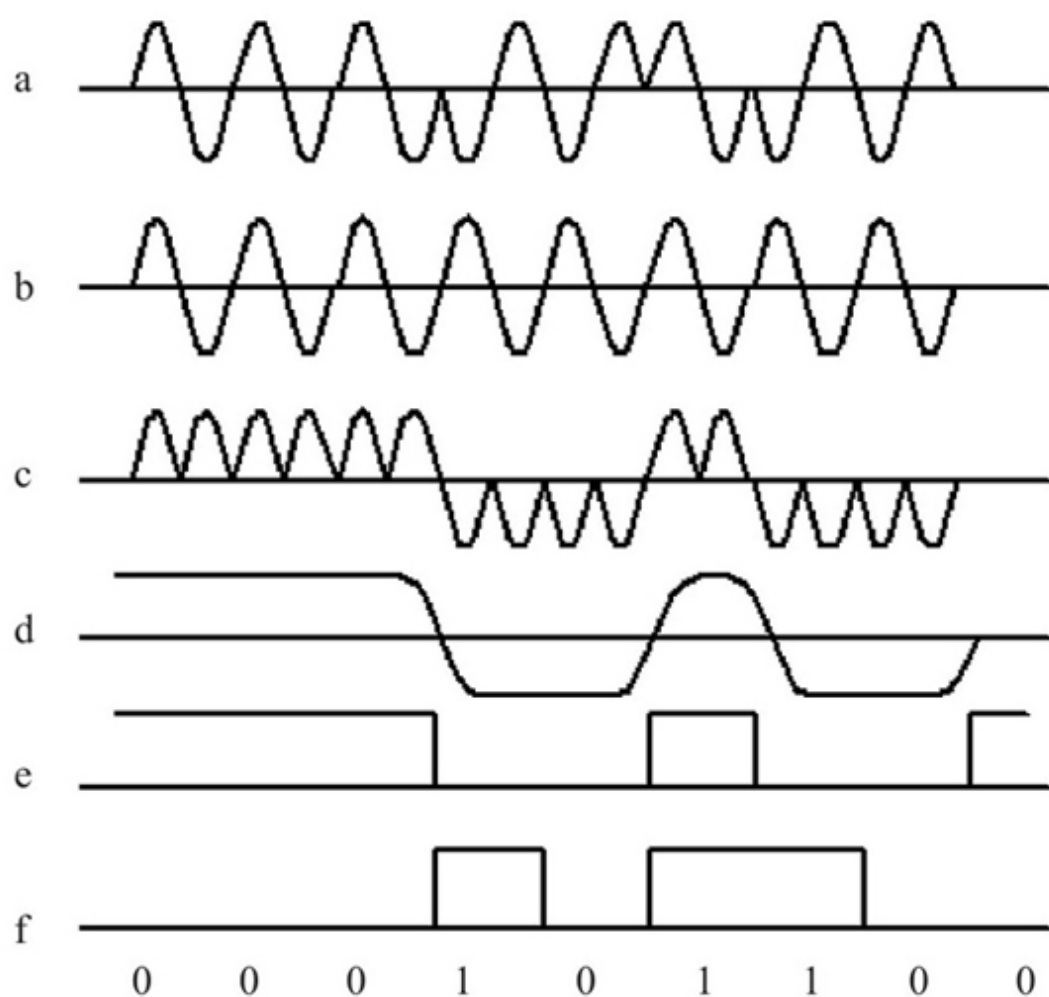


图 4-59 2DPSK相干解调示例

这里从相对码转换成绝对码的计算也是遵从模2逻辑加运算的，运算公式为：

$$a_n = b_n \oplus b_{n-1}$$

式中 b_n 和 b_{n-1} 分别为当前相对码的绝对码，以及前一个相对码的绝对码。计算方法参见前面介绍的2DPSK调制中的绝对码与相对码的转

换，大家可以自己计算验证一下。

因为2DPSK是靠相邻码元的变化与否来决定“1”码和“0”码的，所以不论0相位还是180°相位，相邻码元的变化关系是一样的。也正因如此，接收端无论用0相载波还是180°相位载波解调，尽管得到的差分码不同，但经差分逆变换后，二者得到的结论完全相同。

4.6 物理层传输介质

传输介质又称传输媒介，就是数据传输系统中在发送器和接收器之间的物理线路。在计算机网络中，有多种可用于实际数据传输的传输介质，每一种都有它自己的特性，包括带宽、延迟、工作频率等。这么多传输介质总体上可分为以下两大类：

□导向性传输介质：也就是信号被固定沿着信道的一个或多个方向进行传输的介质，特指有线计算机网络中所使用的传输介质，如双绞线、同轴电缆和光纤等。

□非导向性传输介质：也就是信号在信道中的传输没有固定方向的传输介质，如空气中的无线电波、电磁波，通常是特指在各种无线网络（如WLAN、卫星通信）中所使用的传输介质。

4.6.1 导向性传输介质

通过前面的介绍，我们知道，有线计算机网络中的传输介质都是导向性传输介质，主要有双绞线电缆、同轴电缆和光纤电缆三种。本节向大家简单介绍这三种导向传输介质。

1. 双绞线（Twisted Pair）

双绞线在我们的计机网络，特别是局域网连接中应用最为广泛，这主要得益于它的廉价性和非常不错的可用性。把两根互相绝缘的铜导线并排放在一起，然后用规则的方法绞合起来就构成了双绞线，如图4-60所示。



图 4-60 双绞线

双绞线最早使用在电话交换系统中（那时主要是三类或以前版本的），后来才移植到计算机网络中，当然对应的材质和标准都不一样。在计算机网络中使用的双绞线可分为屏蔽双绞线（Shielded Twisted Pair，STP）和非屏蔽双绞线（Unshielded Twisted Pair，UTP）两大类。STP是以金属箔进行屏蔽包裹的，以减少芯线间的干扰和串音；而UTP双绞线则没有这一层屏蔽用的金属箔，分别如图4-61所示。

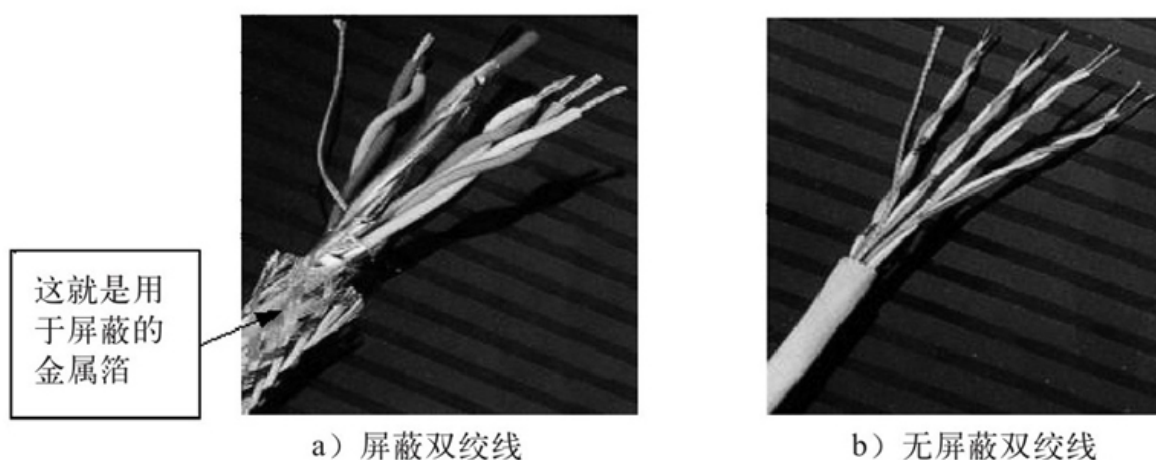


图 4-61 STP和UTP双绞线

除此之外，根据EIA/TIA 568标准规定的双绞线电气性能，计算机网络所用的双绞线还有多种不同的划分，如3类线、4类线、5类线、超5类线、6类线、超6类线，一直到最新的7类线。类型数字越大，版本越新、技术越先进、带宽也越宽，当然价格也越贵了。这些不同类型的双绞线标注方法是这样规定的：如果是标准类型则按“cat×”方式标注，如常用的5类线，则在线的外包皮上标注为“cat5”，注意字母通常是小写，而不是大写。而如果是改进版，就按“×e”，如超5类线就标注为“5e”，同样字母是小写，而不是大写。

双绞线技术标准是在美国通信工业协会（TIA）的EIA/TIA-568A或EIA/TIA-568B标准中定义的，具体如下：

□1类线：是ANSI/EIA/TIA-568A标准中最原始的非屏蔽双绞铜线电缆，但它开发之初的目的不是用于计算机网络数据通信的，而是用于电话语音通信的。

□2类线：是ANSI/EIA/TIA-568A和ISO 2类/A级标准中第一个可用于计算机网络数据传输的非屏蔽双绞线电缆，工作频率为1MHz，传输速率最高为4Mbps。主要用于以前的令牌总线网络。

□3类线：是ANSI/EIA/TIA-568A和ISO 3类/B级标准中专用于10BaseT以太网络的非屏蔽双绞线电缆，工作频率为16MHz，传输速度

最高为10Mbps。

□4类线：是ANSI/EIA/TIA-568A和ISO 4类/C级标准中用于令牌环网络的非屏蔽双绞线电缆，工作频率为20MHz，传输速度最高为16Mbps。

□5类线：是ANSI/EIA/TIA-568A和ISO 5类/D级标准中用于运行CDDI（CDDI是基于双绞铜线的FDDI网络）和快速以太网的非屏蔽双绞线电缆，工作频率为100MHz，传输速度最高为100Mbps。

□超5类线：是ANSI/EIA/TIA-568B.1和ISO 5类/D级标准中用于运行快速以太网的非屏蔽双绞线电缆，工作频率也为100MHz，传输速度最高也为100Mbps。与5类线相比，超5类线在近端串扰、串扰总和、衰减和信噪比这四个主要指标上都有较大的改进。

□6类线：是ANSI/EIA/TIA-568B.2和ISO 6类/E级标准中规定的一种非屏蔽双绞线电缆，主要应用于快速以太网和千兆以太网中。它的工作频率可达200~250MHz，是超5类线带宽的2倍，最高传输速度可达到1000Mbps。

□超6类线：是六类线的改进版，同样是ANSI/EIA/TIA-568B.2和ISO 6类/E级标准中规定的一种非屏蔽双绞线电缆，主要应用于千兆网络中。在工作频率方面与6类线一样，也是200~250MHz，最大传输速

度也可达到1000Mbps，只是在串扰、衰减和信噪比等方面有较大改善。

□7类线：是ISO 7类/F级标准中最新的一种双绞线，主要为了适应万兆以太网技术的应用和发展。但它不再是一种非屏蔽双绞线了，而是一种屏蔽双绞线，工作频率至少可达500MHz，又是6线和超6类线的2倍以上，最高传输速率为10Gbps。

说明 在北美标准中，双绞线及其他一些网络传输介质的标准通常是由以下三个主要的组织制定的：

□ANSI（American National Standard Institute，美国国家标准化组织）；

□TIA（Telecommunication Industry Association，电信工业联合会）；

□EIA（Engineering Institute Association，工程技术协会）。

且通常情况下，其中的一个组织颁布一个标准，再由另外两个组织进行修正。在欧洲，电缆标准由CENELEC（欧洲电子技术标准委员会）发布，该组织的成员来自于欧洲19个国家的电子技术委员会。ISO/IEC将采纳CENELEC的标准作为ISO/IEC的标准。

2.同轴电缆

同轴电缆（Coaxial Cable）的得名也与它的结构相关。它用来传递信息的一对导体是按照一层圆筒式的外导体套在内导体（一根细芯）外面，两个导体间用绝缘材料互相隔离的结构制造的。外层导体和中心轴芯线的圆心在同一个轴心上，所以称为同轴电缆，如图4-62所示。其剖面如图4-63所示。

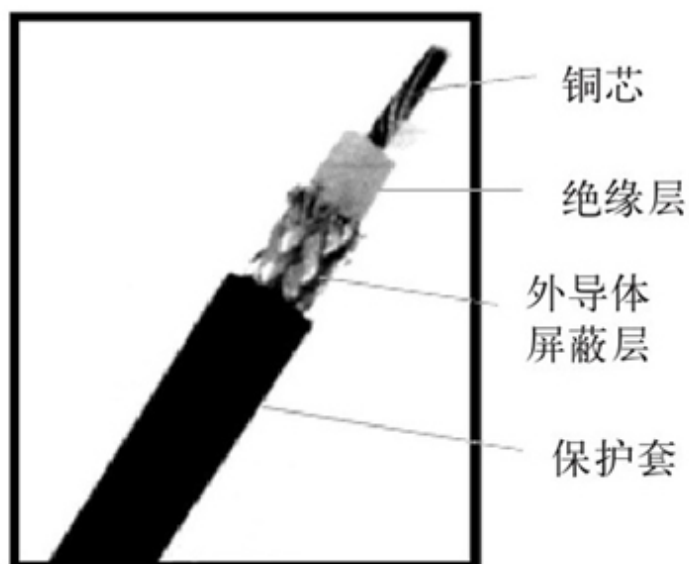


图 4-62 同轴电缆

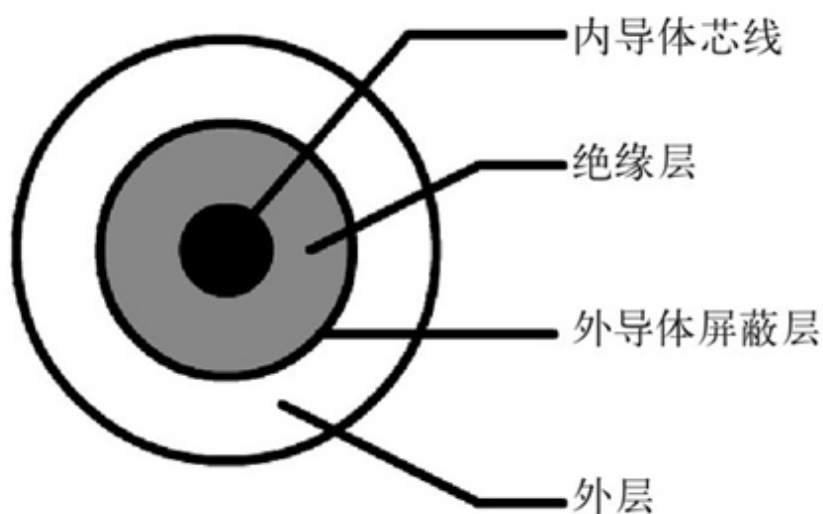
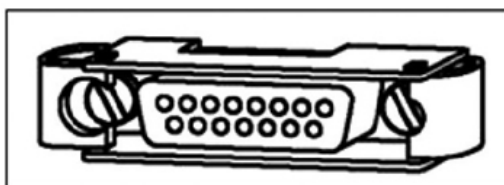


图 4-63 同轴电缆剖面图

广泛使用的同轴电缆有两种：一种为 50Ω 同轴电缆，用于数字信号的传输，即基带同轴电缆；另一种为 75Ω 同轴电缆，用于宽带模拟信号的传输，即宽带同轴电缆。现行以太网同轴电缆的接法有两种：直径为 0.4cm 的RG——11粗同轴电缆，采用凿孔接头接法；直径为 0.2cm 的RG——58细同轴电缆，采用T形头接法。这两种接头如图4-64所示。



a) 粗同轴电缆接头



b) 细同轴电缆接头

图 4-64 粗/细同轴电缆接头

粗同轴电缆符合10BASE5介质标准，使用时需要一个外接收发器（如图4-64a所示）和收发器电缆，单段最大长度为500m，可靠性强，最多可接100台计算机，两台计算机的最小间距为2.5m。细同轴电缆按10BASE——2介质标准直接连到网卡的T形头连接器（即BNC连接器，如图4-64b所示）上，单段最大长度为185m，最多可接30个工作站，最小站间距为0.5m。这些在组网时一定要注意，当然现在用这种电缆组网已基本上没有了。

3.光纤

光纤的完整名称为光导纤维，是用石英玻璃、塑料或晶体等对某个波长范围透明的材料制造的能传输光的纤维。而我们有时听到的“光缆”是含有光纤并符合现场实际使用要求的光、机械和环境规范的线缆，由光纤、加强件和外护层等组成。按光缆中所包含的光纤数量又分为单芯、双芯、多芯几种。图4-65所示即为单芯光缆和多芯光缆结构。

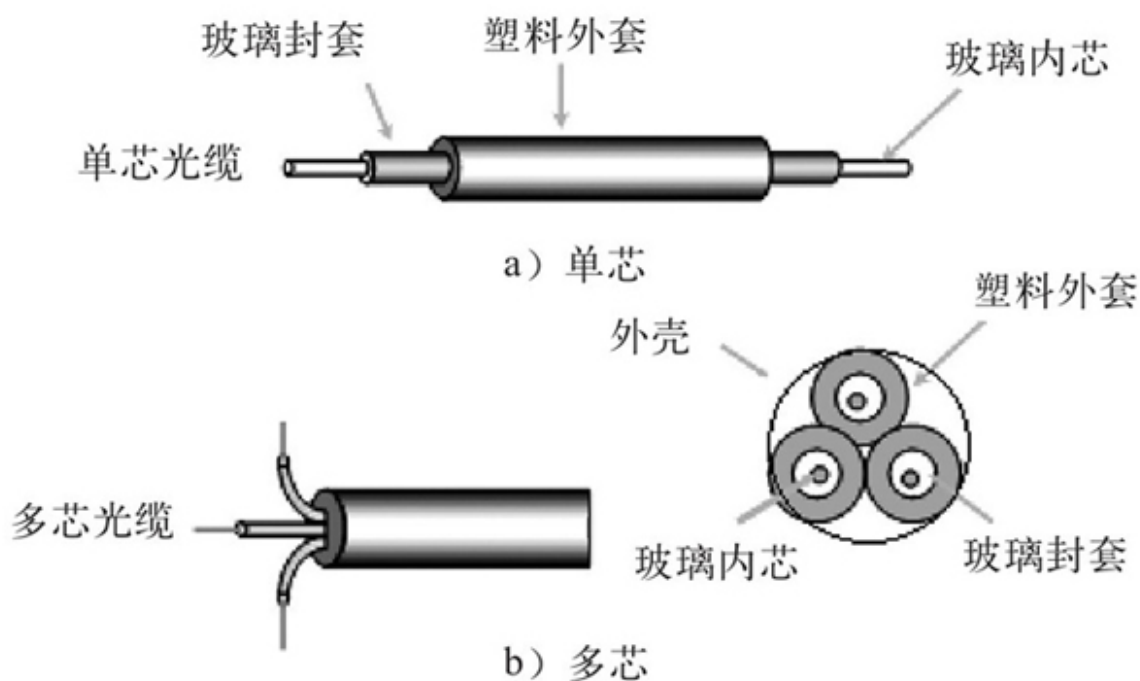


图 4-65 单芯和多芯光缆及其结构

光传输系统由三部分组成，即光源、传输介质和检测器。在光纤的一端放上光源，另一端放上检测器，就构成了一个单向光传输系统。光纤通信是利用光纤传递光脉冲来进行数据通信的。通常是一个光脉冲表示二进制中的一个1，而无光脉冲则表示二进制中的一个0。

光纤的分类标准非常多，主要可以从工作波长、折射率分布、传输模式、原材料和制造方法进行，下面介绍几种主要的分类标准。

(1) 按传输模式来划分

在这里首先要搞清楚的就是什么是光传输模式。光传输模式就是光信号照射到光纤的纤芯以及从光纤包层上反射的角度，一个角度就

是一种模式。根据光纤中传输模式的多少，可分为单模光纤和多模光纤两类。

单模光纤（Single-Mode Fiber, SMF）

单模光纤是一种只用来传输一种模式光（但可以含有多种波长的光）的光纤。也就是在单模光纤中所传输的光照射纤芯的角度和从包层反射的角度是一样的，并且在单模光纤中是没有反射的，只有直射，如图4-66a所示。

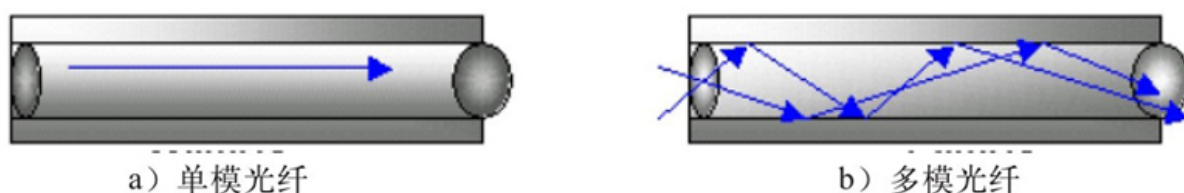


图 4-66 单/双模光纤的信号传输

单模光纤的纤芯直径一般很小（通常为 $8\sim 10\mu\text{m}$ ，光缆外径为 $125\mu\text{m}$ ），因为只有这样光纤才只允许与光纤轴方向一致的光线通过，即只允许通过一个模式的光。单模光纤通常携带一个波长（通常是 $1.31\mu\text{m}$ 或 $1.55\mu\text{m}$ ）的光信号。

因为单模光纤完全避免了反射，所以光信号损耗很小，适合于长距离的光纤通信，如目前在有线电视中采用的就是单模光纤。

多模光纤（Multi-Mode Fiber, MMF）

与单模光纤对应，多模光缆就是可以在一根光纤上同时传输几种模式的光信号的光纤，如图4-66b所示。从图中可以看出，这些光照射纤芯的角度和从包层反射的角度都不一样。

多模光纤的纤芯直径较粗（通常为50 μm 或62.5 μm ，光缆外径也为125 μm ）。由于存在反射，光信号在传输过程中会有损耗，所以多模光纤主要适用于短距离传输，在10Mbps及100Mbps的以太网中传输，而于1Gbps千兆网中，多模光纤最长可支持2000m的传输距离，多模光纤最高可支持550m的传输距离。

（2）按纤芯直径来划分

如果按光纤直径大小来分的话，可以分为：

□ 缓变型多模光纤，50/125（ μm ）；

□ 缓变增强型多模光纤，62.5/125（ μm ）；

□ 缓变型单模光纤，8.3/125（ μm ）。

（3）按光纤芯折射率分布划分

在这种划分标准下，光纤又可以分为以下几种：

阶跃型光纤（Step index fiber, SIF）

阶跃型光纤又称突变型光纤，目前，单模光纤多属此类，最早的多模光纤也属此类。这种SIF光纤纤芯的折射率与包层的折射率成阶跃型分布。它的特点是芯的折射率是均匀的，在芯和包层之间的分界面上，折射率有一个不连续的阶跃性突变。纤芯直径为50~100 μm ，光线以曲折形状传播，特点是信号畸变大。

渐变型光纤（Graded index fiber, GIF）

渐变型光纤又称梯度光纤，单模和多模光纤都有此类型。其特点是纤芯中心的折射率最大，沿径向往外逐步变小，形成一个连续渐变的梯度或坡度，最后达到包层的折射率。渐变型光纤的纤芯直径为50 μm ，光线以正弦形状传播，信号畸变小。

单模环形光纤（Ring fiber）

单模环形光纤折射率分布与突变型光纤相似，纤芯直径为8~12 μm ，光线以直线形状沿纤芯中心轴线方向传播。因为光纤只能传输一个模式，所以称为单模光纤，其折射率最高，信号畸变小。

单模W形光纤（W-Fiber）

单模W形光纤又称双包层光纤，因其折射率分布像W形而得名。它的横截面也分为三个区域，每个区域内的折射率也都是不均匀的，中心的折射率最高，中间区域折射率最低，最外面区域折射率介于两

者之间。可以实现10Gbps容量的100km的超大容量超长距离的信号传送。

单模三角型光纤

单模三角型光纤的纤芯折射率分布曲线为三角形，是一种新型单模光纤，适用于密集波分复用和孤子传输的长距离系统。

单模椭圆型光纤

单模椭圆型光纤也是因其纤芯折射率分布为椭圆型而得名。这种光纤具有双折射特性，即两个正交偏振模的传输常数不同，适用于长距离传输。

(4) 按光纤的组成材料分

按光纤的组成材料可分为：石英玻璃光纤（主要材料为 SiO_2 ）、多组分玻璃光纤（主要材料为 SiO_2 、 Na_2O 和 CaO 等氧化物）、硅酸盐光纤、氟化物光纤、塑料包层玻璃光纤、全塑光纤、液芯光纤、测光光纤、尾光光纤、工业光纤等。光通信中主要用石英光纤，以后所说的光纤也主要是指石英光纤。

(5) 按光纤的套塑层分类

按光纤的套塑层可分为紧套光纤和松套光纤两类。紧套光纤中光纤被套管紧紧地箍住，不能在其中松动，它与塑料套层是一个整体结构。而松套光纤可在套管层中松动。套管内填充油膏，以防水渗入。若一根管内含有多根光纤，则称为松套光纤束。紧套光纤和松套光纤外观及截面结构分别如图4-67所示。

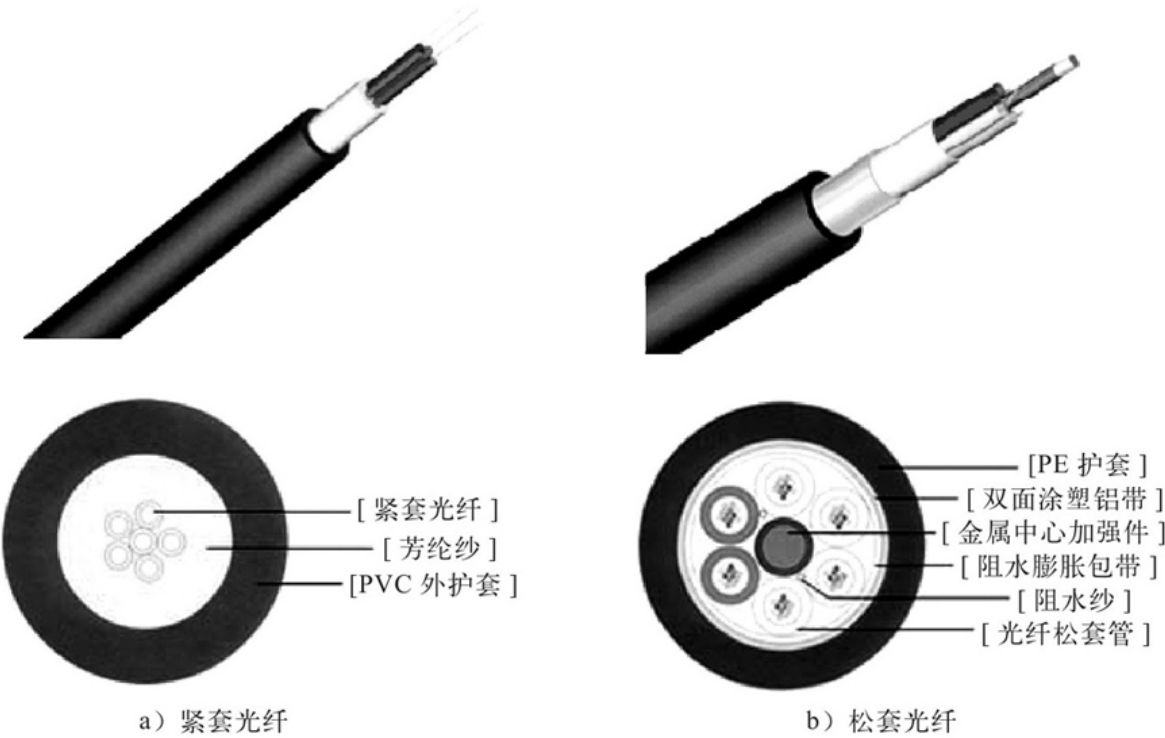


图 4-67 紧套光纤和松套光纤外观及截面结构

4.6.2 光纤结构及主要附件

为了使大家对光纤结构有个初步了解，本节介绍一些常见类型的光纤结构。实际上光纤类型相当复杂，各种不同类型的光纤结构也是多种多样的。

图4-68所示的分别为单芯光纤和多芯光纤截面结构示意。在光纤结构中我们还要了解的一点就是光纤的接口类型，也就是光纤连接器类型。光纤连接器常见的有ST、SC、LC、FC等几类，下面分别予以介绍。

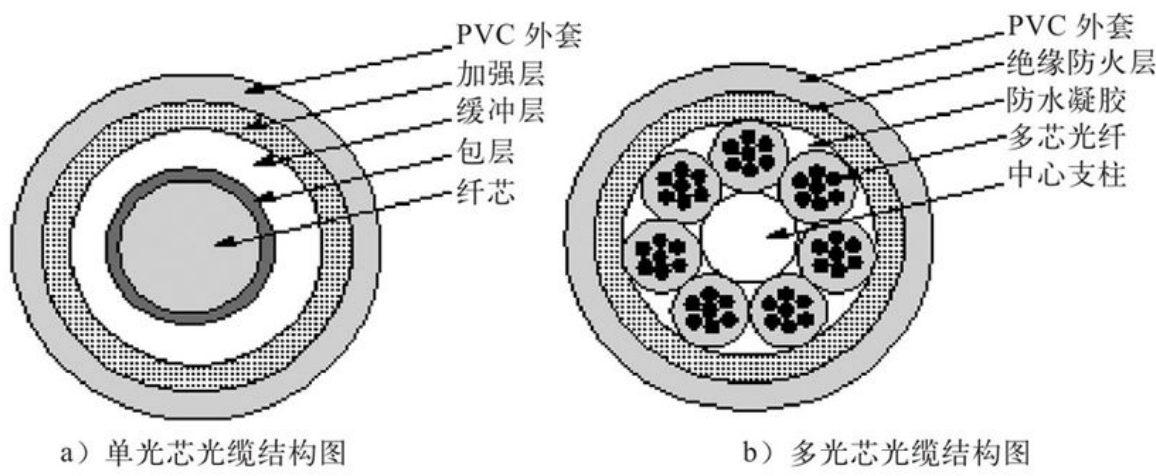


图 4-68 单/多模光纤结构

1. 光纤模块

说到光纤连接器就自然想到常见的两种光纤模块：GBIC（Giga Bitrate Interface Converter，千兆位接口转换器）和SFP（Small Form-factor Pluggable，小型可拔插件），如图4-69所示。GBIC是将千兆位电信号转换为光信号的接口器件。其插入到千兆位以太网端口/插槽内，负责将端口与光纤网络连接在一起，支持热插拔，并在同一个模块上支持IEEE 802.3z标准的1000BaseSX、1000BaseLX/LH或1000BaseZX接口混用。SFP可以简单地理解为GBIC的升级版，体积比GBIC模块减少一半，可以在相同的面板上配置多出一倍以上的端口数量，其他功能基本和GBIC一致。目前更多的是采用SFP模块，因为它体积小，同体积下可以扩展更多的端口数。



图 4-69 GBIC和SFP光纤模块

从图4-69中可以看出，无论是GBIC还是SFP模块，它都会有两个端口（或者偶数个端口），其原因是光纤默认是不支持全双工模式的，它需要一条光纤用于发送数据，一条光纤用于接收数据。不过，目前高档一些的设备都是提供图4-70所示的SFP模块，上面只有一个端

口，接一条光纤。原因就是它采用了WDM（波分复用）技术，使得在一条光纤中可以划分成两个信道，进行全双工传输（模块上那个双向箭头就表示可以双向传输的）。



图 4-70 单端口的全双工SFP模块

前面介绍的GBIC和SFP是两个千兆以太网模块，对于现在万兆（10G）以太网来说，有了新的光纤模块XFP（X就代表“十”的意思），如图4-71所示。XFP模块一般都支持WDM（通常是DWDM，即密集波分复用）技术，所以通常一个模块就接一条光纤，即可实现全双工传输。



图 4-71 XFP光纤模块

2. 光纤连接器

光纤连接器就是接入光模块的光纤接头，其有好多种，且相互之间不可以互用。而且，GBIC和SFP模块的光纤连接器是不同的：SFP模块接LC连接器，GBIC接SC连接器。除了LC、SC连接器外，我们还经常听到另外两种光纤连接器——ST和FC连接器，下面分别予以介绍。

3. ST连接器

ST连接器常用于光纤配线架，光纤芯外露，且接口呈圆形，插针的端面多采用PC型或APC型研磨方式，紧固方式为螺钉扣，如图4-72所示。对于10Base—F光纤以太网标准连接，连接器通常是ST类型的。



图 4-72 ST连接器

前面说的研磨方式是指光纤头的形状，PC（Physical Contact，物理接触）是指纤芯和光纤网络接口的嵌套是平面型的紧密接触，而APC（Angled Physical Contact，呈一定角度的物理接触）是指纤芯和光纤网络接口的嵌套是有一定角度的紧密接触。由于APC的结构与PC完全不同，如果用光纤法兰盘（也就是光纤适配器，是光纤活动连接器对的连接部件，图4-73列出了四种主要的适配器）连接这两种连接器，会损坏连接器的光纤端面。连接APC到PC的办法：通过PC到APC转换的光纤跳线来实现。ST—APC连接器通常是绿色的（黄色的光纤只是单

模光纤），而且人眼就能看到光纤端面的倾斜（也就是有一定角度，不是平面的）。

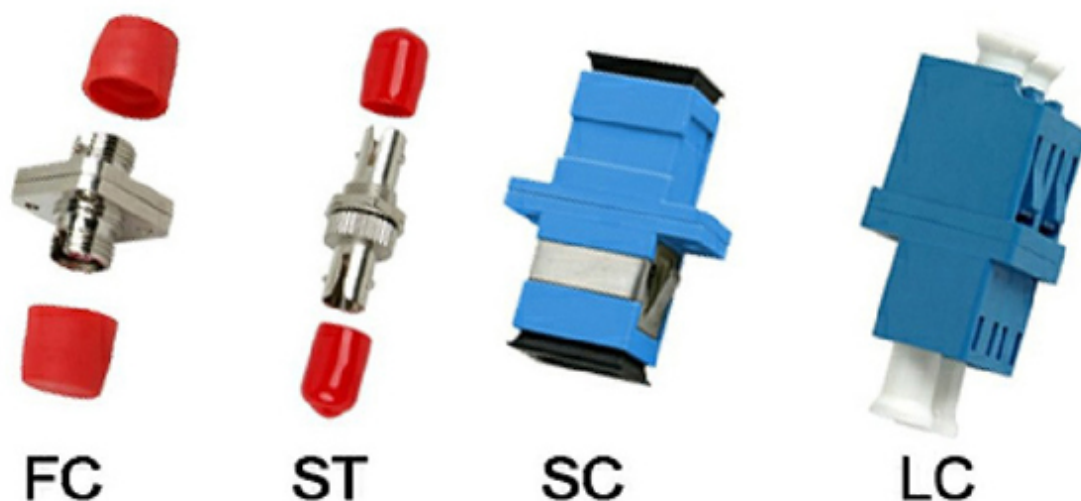


图 4-73 四种主要的光纤适配器

4.SC连接器

SC连接器是由日本NT T公司开发的，是连接GBIC光纤模块的连接器，对于100Base—FX来说，连接器大部分情况下为SC类型的。SC连接器的外壳呈矩形，纤芯在接头里面（ST连接器的纤芯露在外面），其插针的端面多采用PC型或APC型研磨方式（SC—APC连接器通常是绿色的，SC—PC连接通常是蓝色或灰色的），紧固方式采用插拔销闩式，无须旋转，如图4-74所示。此类连接器价格低廉，插拔操作方便，介入损耗波动小，抗压强度较高，安装密度高。

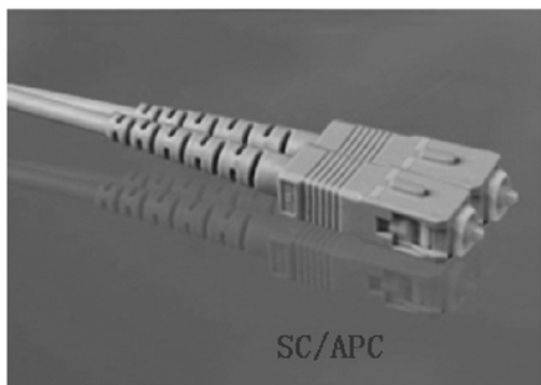


图 4-74 SC连接器

SC连接器在路由器、交换机上用得最多。

5.FC连接器

FC（Ferrule Connector，金属连接器）连接器最早是由日本NT T公司研制，其外部是金属套（连接更加紧），紧固方式为螺钉扣，如图4-75所示。最早的FC连接器采用陶瓷插针的对接端面是平面接触方式，后来采用对接端面呈球面的插针，但外部结构没有改变。

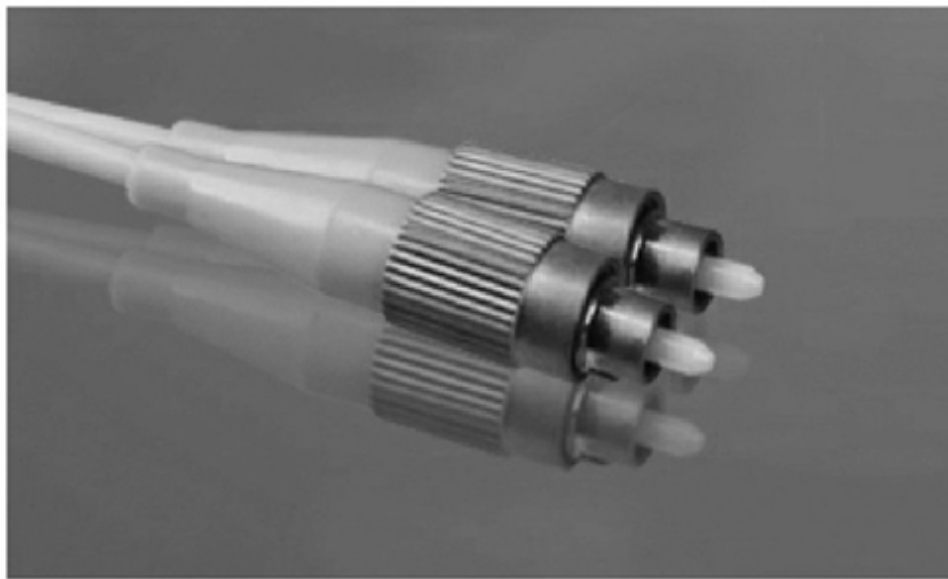


图 4-75 FC连接器

FC连接器一般用于连接配线架。

6.LC连接器

LC连接器是由著名的Bell（贝尔）研究所研究开发出来的，是连接SFP模块的连接器，采用操作方便的模块化插孔(RJ)闩锁机理制成，如图4-76所示。该连接器所采用的插针和套筒的尺寸是普通SC、FC等所用尺寸的一半，提高了光配线架中光纤连接器的密度。

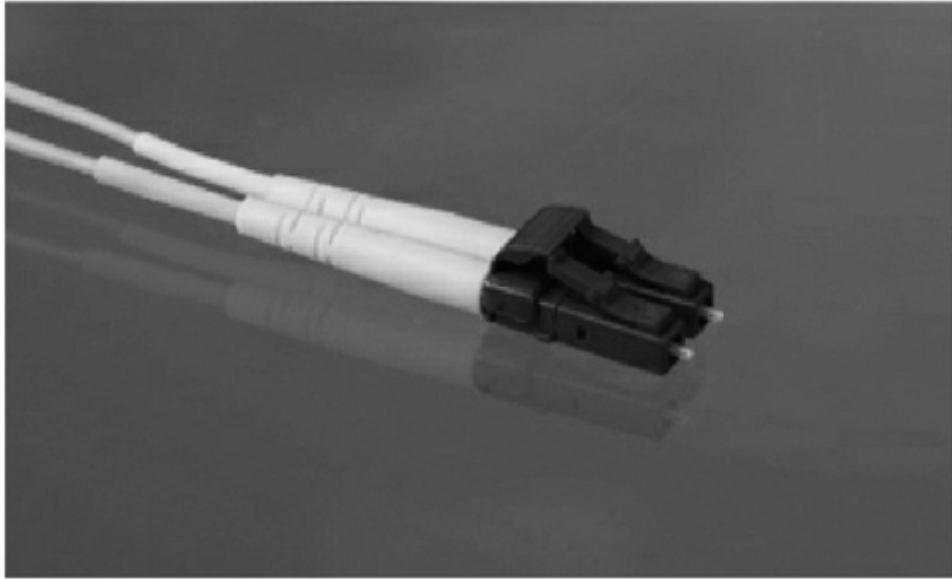


图 4-76 LC连接器

LC连接器在路由器上用得比较多。

光纤两端的连接器可以一样（用于连接相同类型的光纤网络接口），也可以不同（用于连接不同类型的网络接口），图4-77~图4-81是几种常见的带有相同或不同连接器的光纤。



图 4-77 LC-LC光纤



图 4-78 SC-SC光纤



图 4-79 SC-LC光纤



图 4-80 ST-FC光纤



4.6.3 非导向介质

“非导向介质”就是用于无线传输的各种电磁波，在数据通信中主要应用于短波无线传输、地面微波接力通信、地球卫星通信、WLAN（无线局域网）等（其实像蓝牙这类无线通信也是采用非导向介质的）。下面仅简单介绍短波无线传输、地面微波接力通信、地球卫星通信这三种，WLAN在本书第6章有专门的介绍，在此不再赘述。

1.短波无线传输

短波无线传输的实现主要靠电离层的反射。短波的信号频率低于100MHz，但电离层的不稳定所产生的衰落现象和电离层反射所产生的多径效应，使得短波信道的通信质量较差。最典型的短波无线传输应用就是无线局域网连接和手机通信。在这种通信中，固定终端点（基站）和终端之间是通过无线链路连接的。

2.地面微波接力通信

地面微波接力通信在数据通信中占重要地位。微波的频率范围为300MHz~300GHz，在空中主要是直线传播，可经电离反射到很远的地方。同时，由于微波在空中是直线传播，而地球表面是个曲面，因此传输距离受到限制。

地面微波接力通信是在两个地面站之间进行数据传送，距离一般在（50~100）km之间，如图4-82所示。微波接力通信可传输电话、电报、图像、数据等信息。

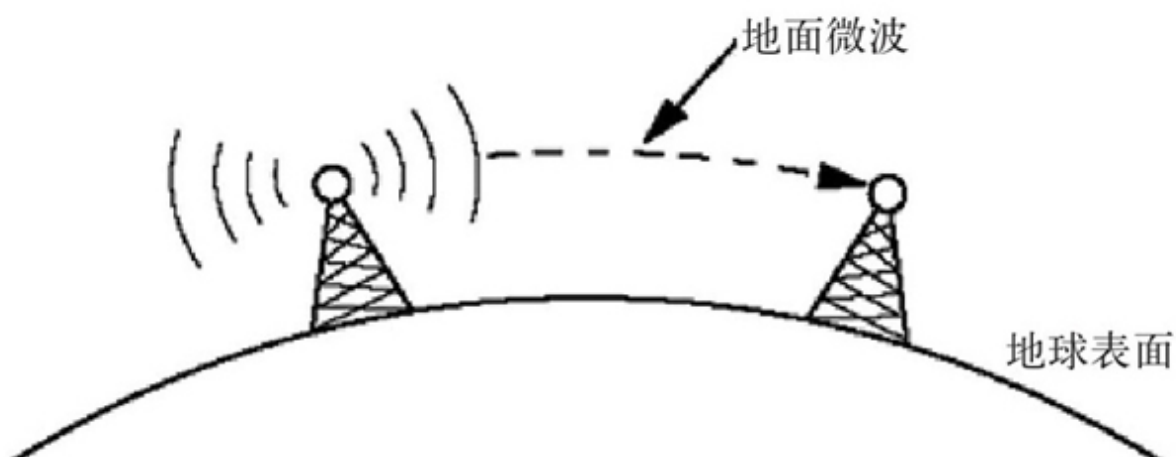


图 4-82 地面微波接力通信

3.卫星通信

很明显，卫星通信就是借助于地球卫星进行的数据通信，如图4-83所示。卫星通信通常采用C波段的（4~6）GHz频段，上行链路为（5.925~6.425）GHz，下行链路为（3.7~4.2）GHz。如果采用KU波段的（12~14）GHz频段，则上行链路为（14~14.5）GHz，下行链路为（11.7~12.2）GHz。

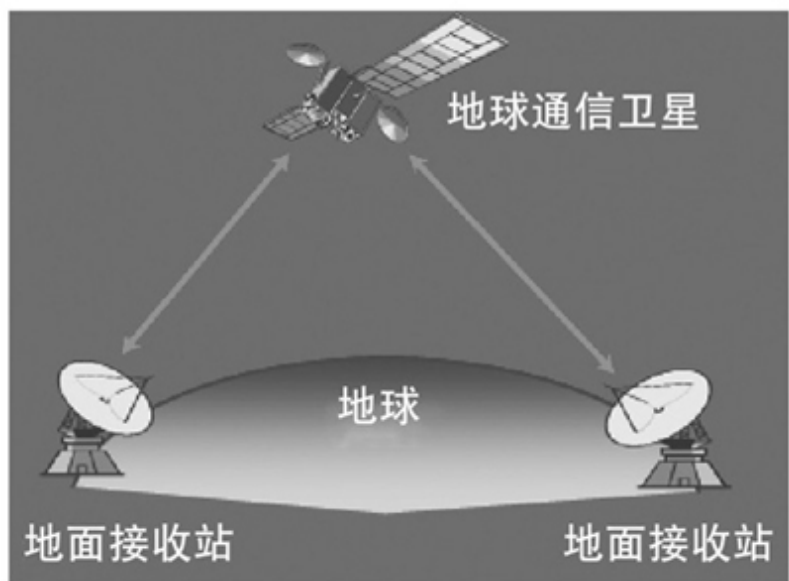


图 4-83 卫星通信示意图

卫星通信的特点：频带宽、通信容量大、通信距离远；且通信费用与通信距离无关；具有较大的传播时延；安全保密性较差，造价高。

4.7 信道多路复用技术

在本章前面已说到，计算机网络中的数据是在物理链路的各个信道中传输的（通常在一条物理链路上会有多条信道的）。默认情况下，一条信道只传输一路信号，这就有可能造成信道带宽资源的浪费，如在某路信号传输所需的带宽严重低于信道带宽时。就像一条大马路上的某条车道，因为该车道在划分时就考虑了大货车通过时对车道宽度的要求，所以车道比较宽。而事实上在这条马路上通过的绝大多数是小汽车，很少有大货车在这条车道中通过。如果仍按交通规则，一条车道只允许一路汽车通过的话，显然就会造成这条大车道利用率的下降。为了解决这个问题，交警可以在这条大车道上再划分成两条小车道，在没有大货车通过时允许两部小汽车在这两条小车道上同时通过，这样可大大提高这条大车道的利用率。

对于计算机网络中的数据传输也一样，我们可以把一条高带宽的信道划分成多条小带宽的子信道（注意，子信道是逻辑意义上的，称为逻辑信道），这样就可以在一条信道上同时进行多路低带宽需求的信号传输，可大大提高原来这条信道的利用率。这就是我们这里所讲的“信道复用”（Channel Multiplexing）技术。这里“复用”就是“共用”，或者“共享”的意思，也就是多路信号共用原来这一条信道进行数据传输。各种信道复用技术都在发送端有对应的“复用器”，而在接收端又

有对应的“分用器”。“分用器”的用途就是把原来在同一信道中传输的各子信道中多路信号分享出来。

根据不同的信号和传输介质类型，可以采用不同的信道复用技术，这些信道复用技术主要有：频分复用（Frequency Division Multiplexing, FDM）、时分复用（Time Division Multiplexing, TDM）、波分复用（Wave Division Multiplexing, WDM）和码分复用（Code Division Multiplexing, CDM）。本节将具体介绍。

4.7.1 频分复用及其原理

频分复用（FDM）是按照信道频带宽度进行信道复用的方式（也是一种调制技术），就是把一条高带宽的信道按照一定的频带宽度划分成若干个低带宽的子频带（或称子信道，如图4-84所示，每个子频带的宽度一般是一样的，但也可以不一样），以实现同一时刻通过不同频率的载波调制技术在信道中传输多路基带信号的目的。如有一个频带宽度（也就是信道带宽）为（1~10）MHz的信道，现在如果要划分成4个子信道，则可划分为（1~2.5）MHz、（2.52~5.0）MHz、（5.02~7.5）MHz、（7.52~10.0）MHz。

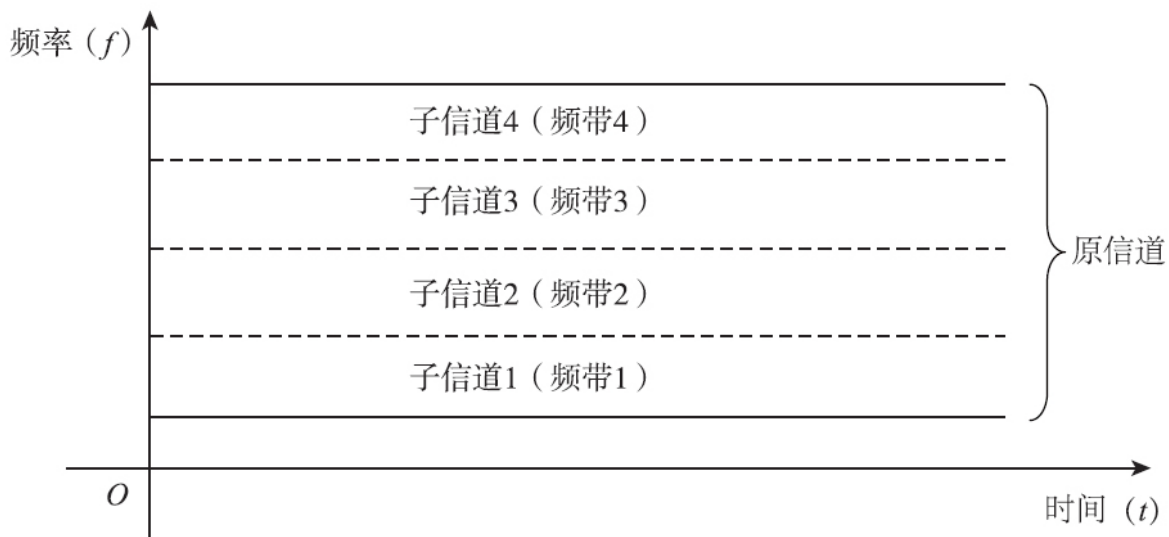


图 4-84 FDM信道复用原理

1.FDM基本原理

在FDM系统中，信道的可用频带被分成若干个互不交叠的频带，从多条线路输入的多路基带信号通过FDM复用器分别用一个不同频带的载波进行调制，然后在同一个信道中的不同子信道中传输，但是每路已调信号只占用原信道中一部分频带（也就是对应子信道中的频带，如图4-85所示），在接收端再通过FDM分用器中的带通滤波器将它们分别滤出来，最后通过各线路的解调装置解调接收即可。

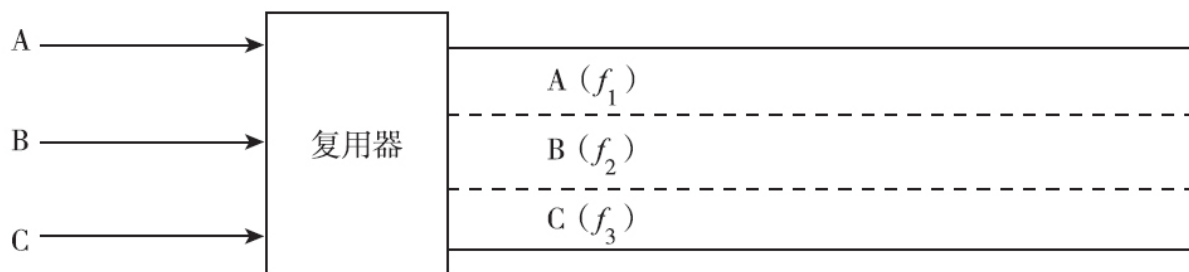


图 4-85 FDM复用示例

这里就涉及到一个信号工作频带的搬移问题，若不搬移在同一信道中传输的多路信号工作频带就可能重叠，若重叠了，自然就达不到信道复用的目的了。这个频带搬移工作就是由FDM复用器来完成的。它可以把进入同一信道的多路信号的工作频带依次搬移到对应子信道的频带中（参见图4-85），这样多路信号就可以在同一个信道中互不干扰地进行传输了。就像一个合唱中，要使每部分人的声音都能很好地被分辨出来，必须要使一部分人采用低音歌唱，一部分人采用中音歌唱，还有一部分人采用高音歌唱。如果大家都采用同一个音调歌唱，就很难区分每部分人唱歌的声音。但在FDM中，为了防止相邻子信道中传输信号的相互干扰，不仅各子信道之间的载波频带是完全分隔、不重叠的，而且相邻子信道中的频带之间最好要有一定间隙，所以FDM对频带的利用率也是相对较低的，当复用的路数不是很多时，可以考虑选择这种信道复用方式。图4-86所示是一个FDM的应用示例，三台主机通过一条线路进行数据传输。

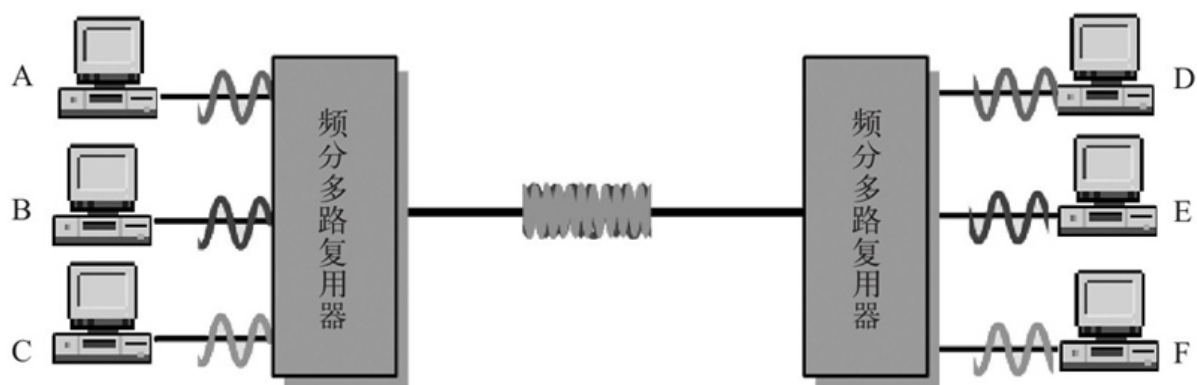


图 4-86 FDM示例

FDM系统的优点是信道复用率高，实现简单，是目前模拟通信中常采用的一种复用方式。但同时**FDM**又存在一些问题，主要表现为各路信号之间存在相互干扰，即串扰。引起串扰的主要原因是滤波器特性不够理想和信道中的非线性特性（就是有突变）造成的已调信号频谱的展宽（也就是一个子信道中的信号频率可能渗透到了另一子信道中）。调制过程中非线性特性所造成的串扰可以一部分一部分地由发送带通滤波器消除，但信道传输中非线性所造成的串扰则无法消除，因而在**FDM**系统对系统线性的要求很高。另外，在调制过程中合理地选择载波频率，并在各路已调信号频谱之间留有一定的保护间隔，也是减小串扰的有效措施。

FDM最典型的应用就是模拟/数字电视、广播系统，电台、电视台使用某个固定特殊的频带或频道来作为某套广播、电视的传输。以前的模拟电话交换系统也是采用**FDM**复用方式的，12路语音信道被调制到载波上各自占据4kHz带宽，可实现在单个物理电路上传输若干条语音信道通信。不过，现在的数字电话系统基本上已被下节将要介绍的**TDM**（时分复用）复用方式所取代了。

2.OFDM

频分复用技术除传统的**FDM**外，还有一种**OFDM**（Orthogonal Frequency Division Multiplexing，正交频分复用）。它是一种多载波调制（(Multi-Carrier Modulation,MCM）技术，可将信道分成若干正交子

信道，然后将高速数据信号转换成并行的低速子数据流，调制到在每个子信道上进行传输。

所谓多载波调制是指发送端将数据流分解为若干个子数据流，从而使子数据流具有低得多的传输比特率，再利用这些数据分别去调制若干个载波，然后同时发送出去的调制技术，其传输过程中采用的是并行化传输技术。与多载波调制对应的就是单载波调制（Single-Carrier Modulation, SCM），它是指将需要传输的数据流调制到单个载波上进行传送的调制技术。如本章前面介绍的ASK、FSK、PSK调制技术都属于单载波调制技术。

“正交”是指各个载波的信号频谱是正交的，也就是各载波的信号频谱可以有交叉，但不能重叠，可以通过某种技术实现完全的分开。由于OFDM使用了无干扰的正交多载波调制技术，不仅提高了多路子信道信号的调制效率，而且由于各载波间可以交叉，无须保护频带，这样使得信道中可用频谱的使用效率更高，比FDM系统中所要求的信道带宽要小得多。另外，OFDM技术可动态分配子信道中的数据，为获得最大的数据吞吐量，多载波调制器可以智能地分配更多的数据到噪声小的子信道上。

目前OFDM技术已被广泛应用于广播式的音频和视频领域以及民用通信系统中，主要的应用包括ADSL、DVB（数字视频广播）、

HDTV（高清晰度电视）、WLAN（无线局域网）和第四代（4G）移动通信系统等。

4.7.2 时分复用及其原理

上节介绍的频分复用（**FDM**）技术是出于单一数据传输过程中可能占用不了整条信道完整带宽，浪费了信道带宽资源的考虑的，这种技术把信道带宽划分成多个小带宽的子信道，把多路信号的发送分别搬移到对应子信道上，以实现信道带宽的更高利用率；而这里的时分复用（**TDM**）则是出于单用户数据传输过程中不可能总有数据传输，通信过程中肯定会存在无数据传输的空闲时段（如用户在输入数据的时候，或者用户在查看数据，或网上信息的时候等）的考虑的，该技术把整个数据通信过程划分成一个个小的时间段（称之为“时间片”），在这些时间片中又划分成多个更小的时间间隙（简称为“时隙”），每个时隙可以用于一个用户信号的传输，这样就可以使得每一个时间片中一般总会有一个或多个时隙中有信号在里面传输（因为一般不太可能多路信号同时没有数据传输），也可以实现提高信道利用率的目的。

如图4-87所示的是一个**TDM**示例，其复用器先把整个数据传输过程划分成一个个小的时间片，然后在这些时间片中又划分了四个时隙，分别用于传输三路信号（相当于三个用户，假设为**A**、**B**、**C**）。

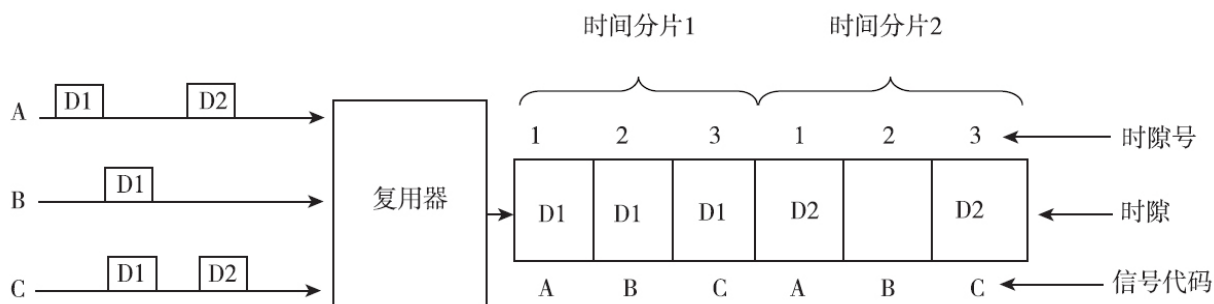


图 4-87 TDM信道复用示意图

假设在某一时间前，A和C路信号发送了2个数据帧（D1和D2），而B路信号只发送了一个数据帧D1，则经过时分复用器复用后，这段时间前发送的5个数据帧分别放在两个时间片中，每个时间片划分了可供三路信号同时传输的三个时隙。此时我们发现在第一个时隙中，A、B、C三路信号都有数据帧D1在传输，而在时间片2中，A和C路信号都在它们各自的时隙中有数据D2传输，而B路信号此时没有数据传输。在接收端通过分复用器即可以还原各路信号。

时分多路复用建立在采样定理基础上，因为采样定理使连续的基带信号变成在时间上离散的采样脉冲，这样，当采样脉冲占据较短时间时，在采样脉冲之间就留出了时间空隙（见图4-88）。利用这种空隙便可以传输其他信号的采样值，因此，就有可能在一条信道同时传送若干个基带信号。时分多路复用以时间作为信号分割的参量，故必须使各路信号在时间轴上互不重叠。由于每路数据总是使用每个时间片的固定时隙，所以这种时分复用又称同步时分复用。

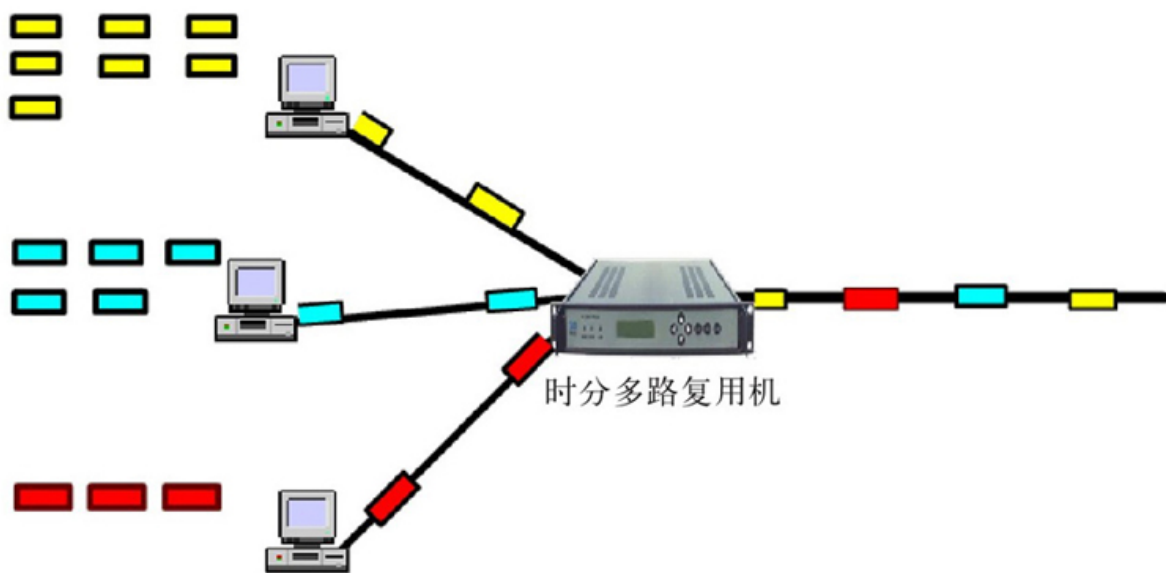


图 4-88 TDM示例

时分复用的典型例子：PCM（Pulse Code Modulation，脉码调制）信号的传输，把多个话路的PCM话音数据用TDM的方法装成帧（帧中还包括了帧同步信息和信令信息），每帧在一个时间片内发送，每个时隙承载一路PCM信号。

通过对以上原理的解析可以看出，尽管TDM通过划分小的时隙可以提高信道的利用率，但还是会存在一些不彻底的问题。如在图4-87中，在第二个时间片中，因B路信号此时没有数据发送，所以该时隙仍处于空闲状态，即使其他路信号的传输很忙，也不能占用该时隙进行数据传输。所以才有了一种改进型的TDM，那就是STDM（Statistical Time Division Multiplexing，统计时分复用）。

STDM是一种可动态分配时隙（也就是不为所有用户分配固定的时隙）的**TDM**。它可以使用按需分配的技术，即根据用户需求动态分配时隙，以避免传统**TDM**中出现空闲时隙的现象，因为**STDM**复用器中的时隙仅为有数据传输的用户分配时隙。同样是因为不为每个用户在每个时间片上分配固定的时隙，所以用户线路的时隙在各时间片中不是周期性出现的，所以**STDM**又称异步时分复用。

与传统**TDM**一样，**STDM**也可与 n 条低速输入用户线路相连。但由于每条输入线路并非一直有数据输入，因此**STDM**时隙数 k 可小于用户线路数 n 。这使得复用信道上的带宽低于各个输入线路带宽之和，或者说对于同样带宽的复用信道，**STDM**可以复接更多的用户线路，即提高了信道的利用率。

4.7.3 波分复用及其原理

波分复用（**WDM**）是光信号中的频分复用技术（前面介绍的**FDM**是针对电信号的），又称光波分复用，是在单条光纤上同时发送多束不同波长激光的技术。因为光波信号的频率一般非常高，不便于表示，而波长相对更便于表示，所以在光信号复用中采用了波分复用这个词。其实它与频分复用是一个意思的，因为不同波长的光波对应的工作频率也不一样。

试想一下，单独拉一条光纤的成本远比单独拉一条双绞线高，如果在一条光纤中只传输一路信号，肯定是不合算的。再加上，不同光纤的波长并不一样，绝大多数是不能完全占用整条光纤的带宽。在以前不采用**WDM**技术的光纤连接中，收、发要用两条光纤，而现在新型的光纤连接中通常都支持**WDM**，只需接一条光纤就可实现收、发两路通信同时双工工作了。

WDM是将两种或多种不同波长的光载波信号在发送端经复用器（又称合波器）汇合在一起，并耦合到光线路的同一根光纤中进行传输的技术。在接收端，经分复用器（又称分波器或去复用器）将各种波长的光载波分离，然后由光接收机做进一步处理以恢复原信号。**WDM**的信道复用原理就是把一个光纤信道中的整个波长带宽划分为若干个小波长范围，每路光载波信号占用一个波长范围来进行传输。这

样一来，原来在一根光纤上只能传送一个光载波的单一信道变为可传送多个不同波长光载波的子信道，从而使得光纤的传输能力成倍增加。如图4-89所示的是三路光纤（A、B、C）中的光载波（假设各自的载波波长分别为 w_1 、 w_2 和 w_3 ）通过合波器后在一条光纤中传输的示例，复用后各路光载波信号的波长带宽分别为 w_4 、 w_5 和 w_6 （对应小于 w_1 、 w_2 、 w_3 ）。

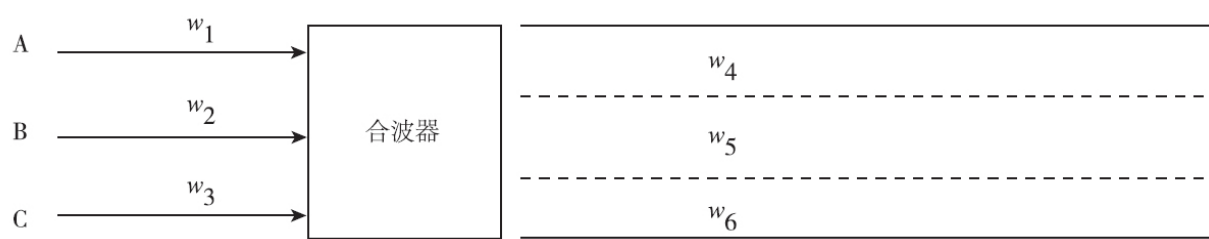


图 4-89 WDM示意图

波分复用一般将波长分割复用器和解复用器（又称合波/分波器）分别置于光纤两端，实现不同光波的耦合与分离。这两个器件的工作原理是相同的。波分复用器的主要类型有熔融拉锥型、介质膜型、光栅型和平面型四种。

波分复用的技术特点与优势如下：

（1）灵活增加光纤传输容量

波分复用技术可充分利用光纤的低损耗波段，增加光纤的传输容量，使一根光纤传送信息的物理限度增加一倍至数倍。对已建光纤系

统，尤其早期铺设的芯数不多的光缆，只要原系统有功率余量，可进一步增容，实现多个单向信号或双向信号的传送而不用对原系统做大改动。具有较强的灵活性。

（2）同时传输多路信号

波分复用技术使得在同一根光纤中传送2个或多个非同步信号成为可能，有利于数字信号和模拟信号的兼容。而且与数据速率和调制方式无关，在线路中间可以灵活取出或加入信道。

（3）成本低、维护方便

采用波分复用技术可大量减少光纤的使用量，进而大大降低建设成本。由于光纤数量少，当出现故障时，恢复起来也迅速方便。

（4）可靠性高，应用广泛

由于系统中有源设备大幅减少，这样就提高了系统的可靠性。目前由于多路载波的波分复用对光发射机、光接收机等设备要求较高，技术实施有一定难度。同时多纤芯光缆的应用对于传统广播电视传输业务未出现特别紧缺的局面，因而WDM的实际应用还不多。但是随着有线电视综合业务的开展，网络带宽需求日益增长，出于对各类选择性服务的实施、网络升级改造经济费用的考虑等，光波复用的特点和

优势在CATV传输系统中逐渐显现出来，表现出广阔的应用前景，甚至将影响CATV网络的发展格局。

4.8 物理层接口

ISO对OSI模型的物理层所做的定义为：在物理信道实体之间合理地通过中间系统，为比特传输所需的物理连接的激活、保持和去除提供机械的、电气的、功能性和规程性的手段。比特流传输可以采用异步传输，也可以采用同步传输完成。

另外，CCITT在X.25建议书第一级（物理级）中也做了类似的定义：利用物理的、电气的、功能的和规程的特性在DTE和DCE之间实现对物理信道的建立、保持和拆除功能。这里的DTE（Date Terminal Equipment）指的是数据终端设备，是对属于用户所有的连网设备或工作站的统称，它们是通信的信源或信宿，如计算机、终端等；DCE（Date Circuit Terminating Equipment或Date Communications Equipment），指的是数据电路终接设备或数据通信设备，是对为用户提供接入点的网络设备的统称，如自动呼叫应答设备、调制解调器等。

4.8.1 串行接口标准

在计算机网络数据通信中，有几个接口标准是经常见到和用到的，那就是通常所说的RS-232、RS-422与RS-485标准。它们都是串行

数据接口标准，最初都是由电子工业协会（EIA）制定并发布的。

RS-232接口（又称EIA RS-232）是目前最常用的一种串行通信接口。它是在1970年由美国电子工业协会（EIA）联合贝尔系统、调制解调器厂家及计算机终端生产厂家共同制定的用于串行通信的标准。它的全名是“数据终端设备（DTE）和数据通信设备（DCE）之间串行二进制数据交换接口技术标准”。该标准规定采用一个25个脚的DB25连接器，对连接器的每个引脚的信号内容进行了规定，还对各种信号的电平进行了规定。

RS-422由RS-232发展而来，它是为弥补RS-232之不足而提出的。为改进RS-232通信距离短、速率低的缺点，RS-422定义了一种平衡通信接口，将传输速率提高到10Mbps，传输距离延长到4000英尺（速率低于100kbps时），并允许在一条平衡总线上连接最多10个接收器。RS-422是一种单机发送、多机接收的单向、平衡传输规范，后来被命名为TIA/EIA-422A标准。为扩展应用范围，EIA又于1983年在RS-422基础上制定了RS-485标准，增加了多点、双向通信能力，即允许多个发送器连接到同一条总线上，同时增加了发送器的驱动能力和冲突保护特性，扩展了总线共模范围，后命名为TIA/EIA-485A标准。由于EIA提出的建议标准都是以“RS”作为前缀，所以在通信工业领域，仍然习惯将上述标准以RS作前缀，也就是上面说到的三个串行接口标准。

RS-232、RS-422与RS-485标准只对接口的电气特性做出规定，而不涉及接插件、电缆或协议，在此基础上用户可以建立自己的高层通信协议。因此在视频界的应用，许多厂家都建立了一套高层通信协议，或公开或厂家独家使用。

表4-1所示是以上三种串行接口电气性能参数的综合比较。

表 4-1 RS-232、RS-422 与 RS-485 串行接口电气性能参数综合比较

性能参数	RS-232	RS-422	RS-485
工作方式	单端	差分	差分
节点数	1 收、1 发	1 发、10 收	1 发、32 收
最大传输电缆长度 / 英尺	50	400	400
最大传输速率 /bps	20k	10M	10M
最大驱动输出电压 /V	+/-25	-0.25 ~ +6	-7 ~ +12
驱动器输出信号电平（负载最小值）/V	+/-5 ~ +/-15	+/-2.0	+/-1.5
驱动器输出信号电平（空载最大值）/V	+/-25	+/-6	+/-6
驱动器负载阻抗 /Ω	3k ~ 7k	100	54
摆率（最大值）	30V/μs	N/A	N/A
接收器输入电压范围 /V	+/-15	-10 ~ +10	-7 ~ +12
接收器输入门限	+/-3V	+/-200mV	+/-200mV
接收器输入电阻 /Ω	3k ~ 7k	4k（最小）	≥ 12k
驱动器共模电压 /V		-3 ~ +3	-1 ~ +3
接收器共模电压 /V		-7 ~ +7	-7 ~ +12

4.8.2 RS-232串行接口标准

目前RS-232是PC机与通信工业中应用最广泛的一种串行接口，其中RS（ecommeded standard）代表推荐标准，232是标识号。RS-232被定义为一种在低速率串行通信中增加通信距离的单端标准。RS-232采取不平衡传输方式，即所谓单端通信。一个完整的RS-232接口有22根线，采用标准的25芯插头座（DB-25）。除此之处，目前广泛应用的还有一种9芯的RS-232接口（DB-9）。它们的外观都是D形的，不过，对接的两个接口又分为针式的公头和孔式的母头两种，DB-9的母头和公头和DB-25的母头和公头分别如图4-90所示。

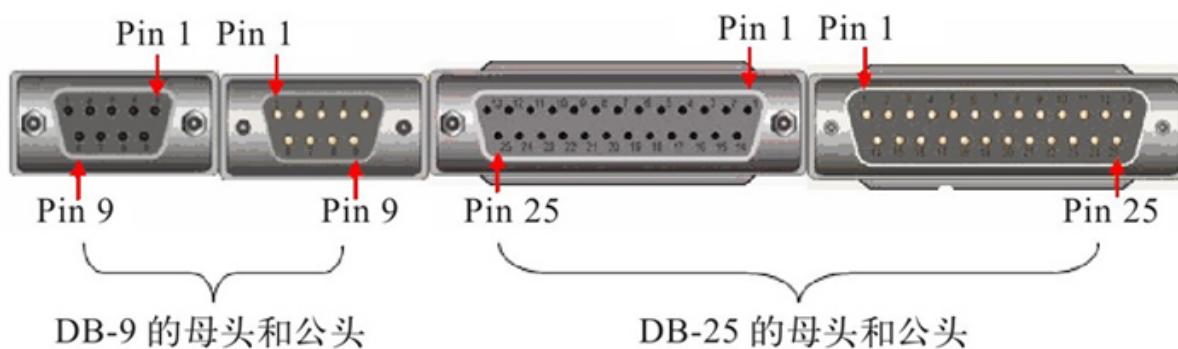


图 4-90 DB-9/DB-25的公头和母头

在RS-232标准中定义了逻辑一和逻辑零电压级数，以及标准的传输速率和连接器类型。信号大小在 $\pm 3 \sim \pm 15V$ 之间。RS-232规定接近零的电平是无效的，逻辑一规定为负电平，有效负电平的信号状态称为

传号（marking），它的功能意义为OFF；逻辑零规定为正电平，有效正电平的信号状态称为空号（spacing），它的功能意义为ON。根据设备供电电源的不同， $\pm 5V$ 、 $\pm 10V$ 、 $\pm 12V$ 和 $\pm 15V$ 这样的电平都是可能的。

1.RS-232连接器

RS-232标准中规定的设备可以分为数据终端设备（DTE）和数据通信设备（DCE）两类，这种分类定义了用不同的线路来发送和接受信号。一般来说，计算机和终端设备用DTE连接器，调制解调器和打印机用DCE连接器。

RS-232标准中指定了20个不同的信号连接，而RS-232连接器是由25个D-sub（微型D类）引脚构成的DB-25连接器。但很多设备只是用了其中的一小部分引脚，出于节省资金和空间的考虑，不少机器采用较小的连接器，特别是9引脚的D-sub或者是DB-9型连接器被广泛应用于绝大多数IBM的AT机之后的PC机和其他许多设备上。

因为RS-232到目前为止经历了好多个不同版本，最近的版本号为E，它相对目前广泛应用的C版本来说，电气性能改进了不少，也对连接器中的25个引脚进行了充分利用（只有2个予以保留）。表4-2列出的是25芯RS-232E接口的信号和引脚分配，而表4-3所示的则是在PC机、

调制解调器和路由器等网络设备中使用的9芯RS-32接口的信号和引脚分配。

表 4-2 25 芯 RS-232 接口的信号和引脚分配

引 脚 号	缩 写 符	信号方向	说 明
1	PE	公共端	连接器外壳
2	TXD	输出	发送数据
3	RXD	输入	接收数据
4	RTS	输出	请求发送
5	CTS	输入	允许发送
6	DSR	输入	数据设备准备好
7	SG	公共端	信号地
8	DCD	输入	载波检测
9	—	—	保留
10	—	—	保留
11	STF	输出	选择传送通道
12	DCD	输入	载波检测
13	CTS	输入	允许发送
14	TXD	输出	发送数据
15	TCK	输入	发送时钟
16	RXD	输入	接收数据

(续)

引 脚 号	缩 写 符	信号方向	说 明
17	RCK	输入	接收器定时
18	LL	输出	本地回路控制
19	RTS	输出	请求发送
20	DTR	输出	数据终端准备好
21	RL	输出	远程回路控制
22	RI	输入	振铃指示器
23	DSR	—	数据装置准备好
24	XCK	输出	发送器定时
25	TI	输入	测试指示器

表 4-3 9 芯 RS-232 接口的信号和引脚分配

引 脚 号	缩 写 符	信号方向	说 明
1	DCD	输入	载波检测
2	RXD	输入	接收数据
3	TXD	输出	发送数据
4	DTR	输出	数据终端准备好
5	GND	公共端	信号地
6	DSR	输入	数据装置准备好
7	RTS	输出	请示发送
8	CTS	输入	清除发送
9	RI	输入	振铃指示

在RS-232C版本中，DB-25的25个引脚（4个数据引脚、11个控制信号引脚、3个定时引脚、7个备用和未定义引脚）实际上只用了表4-2中的9个，分别是1、2、3、4、5、6、8、20、22引脚。它们的作用分别如下：

（1）控制信号引脚（4、5、6、8、20和22引脚）

□数据装置准备好（Data set ready，DSR）：第6脚，有效时（ON）状态，表明接口处于可以使用的状态。

□数据终端准备好（Data set ready，DTR）：第20脚，有效时（ON）状态，表明数据终端可以使用。

这两个信号有时连到电源上，一上电就立即有效。但这两个设备状态信号有效，只表示设备本身可用，并不说明通信链路可以开始进行通信了，能否开始进行通信要由下面的控制信号决定。

□请求发送（Request to send, RTS）：第4脚，用来表示DTE请求DCE发送数据，即当终端要发送数据时，使该信号有效（ON状态），向DTE设备请求发送。

□允许发送（Clear to send, CTS）：第5脚，用来表示DCE准备好接收DTE发来的数据，是对请求发送信号RTS的响应信号。

当Modem之类设备已准备好接收终端传来的数据并向前发送时，使CTS信号有效，通知终端开始沿发送数据线TxD发送数据。这对RTS/CTS请求应答联络信号用于半双工Modem系统中发送方式和接收方式之间的切换。在全双工系统中，因配置双向通道，故不需要RTS/CTS联络信号。

□数据载波检出（Data Carrier detection, DCD）：第8脚，又称接收线信号检出（Received Line Detection, RLSD），用来表示DCE已接通通信链路，告知DTE准备接收数据。当本地的Modem收到由通信链路另一端（远地）的Modem送来的载波信号时，使RLSD信号有效，通知终端准备接收，并且由Modem将接收下来的载波信号解调成数据信号后，沿接收数据线RxD送到终端。

□振铃指示（Ringing, RI）：第22脚，当Modem收到交换台送来的振铃呼叫信号时，使该信号有效（ON状态），通知终端，已被呼叫。

（2）数据发送与接收线（2和3引脚）

□发送数据（Transmitted data, TxD）：第2脚，通过TxD终端将串行数据发送到Modem（DTE → DCE）。

□接收数据（Received data, RxD）：第3脚，通过RxD线终端接收从Modem发来的串行数据（DCE → DTE）。

（3）地线（1引脚）

SG（7脚）、PE（1脚）分别用来接信号地和保护地信号线，无方向。

上述控制信号线有效、无效的顺序表示了接口信号的传送过程。例如，只有当DSR和DTR都处于有效（ON）状态时，才能在DTE和DCE之间进行传送操作。若DTE要发送数据，则预先将DTR线置成有效（ON）状态，等CTS线上收到有效（ON）状态的回答后，才能在TxD线上发送串行数据。这种顺序的规定对半双工的通信线路特别有用，因为半双工的通信只有确定DCE已由接收方向改为发送方向，这时线路才能开始发送。

使用DB-9连接器，作为提供多功能I/O卡或主板上COM1和COM2两个串行接口的连接器。它只提供异步通信的9个信号。DB-25型连接

器的引脚分配与DB-25型引脚信号完全不同。因此，若与配接DB-25型连接器的DCE设备连接，必须使用专门的电缆线。

在连接距离上，通信速率低于20kbps时，RS-232C直接连接的最大物理距离为15m（50英尺）。

2.工作原理

信号的标注是从DTE设备的角度出发的，TD、DTR和RTS信号是由DTE产生的，RD、DSR、CTS、DCD和RI信号是由DCE产生的。接地信号是所有连接公用的，在有的标准中接地信号外部有两个引脚，事实上是同一个信号。如果两个通信设备的距离相差的很远或者有两个不同的供电系统供电，那么地信号在两个设备间会不一样，从而导致通信失败，跟踪描述这样的情形是很困难的。

收、发端的数据信号是相对于信号地的。如从DTE设备发出的数据在使用DB-25连接器时是2脚相对7脚（信号地）的电平。典型的RS-232信号在正负电平之间摆动，在发送数据时，发送端驱动器输出正电平在+5~+15V之间，负电平在-5~-15V之间。当无数据传输时，线上为TTL，从开始传送数据到结束，线上电平从TTL电平到RS-232电平再返回TTL电平。接收器典型的工作电平在+3~+12V与-3~-12V之间。由于发送电平与接收电平的差仅为2V至3V，所以其共模抑制能力差，再加上双绞线上的分布电容，其传送距离最大为约15m，最高速率为

20kbps。RS-232是为点对点（即只用一对收、发设备）通信设计的，其驱动器负载为（3~7）k Ω 。所以RS-232适合本地设备之间的通信。

注意 RS-232C标准中提到的发送和接收，都是站在DTE立场上，而不是站在DCE的立场来定义的。由于在计算机系统中，往往是CPU和I/O设备之间传送信息，两者都是DTE，因此双方都能发送和接收。

4.8.3 其他EIA标准接口

除了前面介绍的EIA-232（即RS-232）接口外，在EIA接口标准中，常见的串行接口协议还有EIA-422、EIA-449、EIA-485、EIA-530等，本节予以简单介绍。

EIA-422（过去称为RS-422）是采用4线、全双工、差分传输、多点通信的数据传输接口协议。与下面将要介绍的EIA-485不同的是，EIA-422不允许出现多个发送端，只能有多个接收端（EIA-232接口也可以有多个接收端）。EIA-232型可以有两端口、四端口和八端口等几种型号，而EIA-422型接口可以有四端口、八端口，甚至十端口等几种型号。

EIA-422传输信号的距离和速度比EIA-232更远、更快，并且一般都能抗电子干扰和电涌。当电缆线的长度为12m（40码）时传输速率可以达到10Mbps。由于EIA-422具有上述优点，因此成为工业、制造业以及分布广泛的销售经营企业的首选产品。

EIA-422的通常用途是作为RS-232的扩展，是RS-232的一个变种。EIA-422作为mini-DIN-8连接器标准，在被USB连接器取代前，曾在苹果麦金托什（MAC）电脑上大量使用。

EIA-422接口的机械特性由EIA-530或EIA-449规定，然而设备仅在发送方和接受方成对出现时才通信。电缆的最高传输速率在长度为1.2m时为10Mbps，为1200m时100kbps。EIA-422不能实现像EIA-485那样的真正的多点通信，尽管其一个发送端就可以连接最多10个接收端。

EIA-449（过去称为RS-449）接口协议是于1992年9月发布的。它规定了数据终端设备和数据通信设备之间的接口的功能特性和机械特性。规定RS-449采用平衡传输时的电气特性的协议是RS-422。规定非平衡传输时的电气特性的协议是RS-423，数据的传输率可达200kbps。协议规定了两个D-sub连接时第一个为37引脚，第二个9引脚。尽管这种协议没有在个人电脑上使用，但在大型数据交换服务器上还是很常见的。

EIA-485（过去称为RS-485或RS485）接口协议是电力特性规定为2线、全双工、多点通信的标准。它的电力特性和RS-232大不一样。用缆线两端的电压差值来表示传递信号。1端的电压标识为逻辑1，另一端标识为逻辑0。两端的电压差为0.2V以上时有效，任何不大于12V或者不小于-7V的差值对接收端来说都是正确的。

EIA-485仅规定了接收端和发送端的电力特性，没有规定或推荐任何数据协议。

EIA-485可用于配置便宜的广域网和可采用单机发送、多机接收的通信联结，可提供高速的数据通信速率（电缆长10m时35Mbps，电缆长200m时100kbps）。EIA-485和EIA-422一样使用双绞线进行高电压差分平衡传输，可以进行大面积长距离传输（最长为1200m）。与EIA-422不同的是，EIA-422采用不可转换的单发送端，EIA-485的发送端需要设置为发送模式，这使得EIA-485可以使用双线模式实现真正的多点双向通信。

EIA-485推荐使用在点对点网络中，如线型、总线型拓扑结构的网络，但不能是星型、环型网络。理想情况下EIA-485需要2个终接电阻，其阻值要求等于传输电缆的特性阻抗。没有特性阻抗的话，当所有的设备都静止或者没有能量的时候就会产生噪声。没有终接电阻的话，会使得较快速的发送端产生多个数据信号的边缘，这其中的一些是不正确的。EIA-485不能用在星型或者环型的拓扑结构中，主要是由于在这些结构中，EIA-485接口中过低或者过高的终接电阻会产生电磁干扰。

EIA-485在使用4线时可以和EIA-422一样实现全双工，还可实现真正的多点通信。在某些限制条件下EIA-485和EIA-422可以实现相互的连接。如SCSI-2和SCSI-3控制卡通常使用这种标准的设备来作为物理层。EIA-485接口同样可以在一些工厂的项目控制机器上看到，来实现

工厂不同楼层之间的数据通信。它可以抵抗机械设备和焊接设备的电磁干扰。

EIA-485还可以在大型音频系统中使用，如在音乐厅和剧院中通过这种接口的设备就可以使用普通的计算机来运行一些特殊的软件以实现远距离音频设备的控制。EIA-485通过XLR标准的线缆连接的设备大量用于麦克风上，从而实现舞台和控制台之间的连接而无须预设线路。

表4-4列出了EIA-485的一些特性、引脚的分配以及和RS-232的比较。

表 4-4 EIA-485 的一些特性、引脚的分配以及和 RS-232 的比较

EIA-485	RS-232	DB-25	DE-9
Common Ground	Carrier Detect (DCD)	8	1
Clear To Send + (CTS+)	Received Data (RD)	3	2
Ready To Send + (RTS+)	Transmitted Data (TD)	2	3
Received Data + (RxD+)	Data Terminal Ready (DTR)	20	4
Received Data - (RxD-)	Common Ground	7	5
Clear To Send - (CTS-)	Data Set Ready (DSR)	6	6

(续)

EIA-485	RS-232	DB-25	DE-9
Ready To Send - (RTS-)	Request To Send (RTS)	4	7
Transmitted Data + (TxD+)	Clear To Send (CTS)	5	8

4.8.4 X.21、X.24、X.36和EIA-530接口规范

除了在前面介绍的RS-232物理层接口外，目前在广域网的网络连接中，常用的其他物理接口还有X.21、X.24、X.36和EIA-530这四种。本节将简单对这些物理接口规范进行介绍。

X.21一开始是CCITT在1976年以建议的形式向外发布的，它定义了用户计算机的DTE如何与数字化的DCE交换信号的数字接口标准。X.21的设计目标之一是要减少RS-232之类的串行接口中的信号线的数目，其采用15芯标准连接器代替原来的25芯连接器，而且其中仅定义了8条接口线。

X.21的另外一个设计目标是允许接口在比EIA RS-232C更长的距离上进行更高速率的数据传输，其电气特性类似于EIA RS-422的平衡接口，支持最大的DTE-DCE电缆距离是300m。X.21可以按同步传输的半双工或全双工方式运行，传输速率最大可达10Mbps。X.21接口适用于通过数字线路（而不是模拟线路）访问公共数据网（PDN）的地区。欧洲网络大多使用X.21接口。

X.21接口的15针连接器中定义了8条功能线。分成4个工作阶段：空闲，呼叫控制，数据传送，清除。X.21接口的接口规范如表4-5所

示。X.24 DTE接口规范、X.24 DCE接口规范、X.36和EIA-530接口规范分别如表4-6、表4-7、表4-8、表4-9所示。

表 4-5 X.21 接口规范

引 脚 号	缩 写 符	信号方向	说 明
1	PG	公共端	设备地
2	T (A)	输出	发送数据 (+)
3	C (A)	输出	控制信号 (+)
4	R (A)	输入	接收数据 (+)
5	I (A)	输入	指示 (+)
6	S (A)	输入	信号元素计时 (+)
7	B (A)	输入	字节计时 (+)
8	SG	公共端	信号地
9	T (B)	输出	发送数据 (-)
10	C (B)	输出	控制信号 (-)
11	R (B)	输入	接收数据 (-)

(续)

引 脚 号	缩 写 符	信号方向	说 明
12	I (B)	输入	指示 (-)
13	S (B)	输入	信号元素计时 (-)
14	B (B)	输入	字节计时 (-)

表 4-6 V.24DTE 接口规范

引 脚 号	缩 写 符	信号方向	说 明
1	PG	保护接地	公共端
2	TXD	输出	发送数据
3	RXD	输入	接收数据
4	RTS	输出	请示发送
5	CTS	输出	清除发送
6	DSR	输入	数据设备准备好
7	SG	公共端	信号地
8	DCD	输出	载波检测
15	RCLK	输入	接收时钟
17	RCLK	输入	接收时钟
20	DTR	输出	数据终端准备好
24	DTECLK	输出	发送时钟 (DTE)

表 4-7 V.24 DCE 接口规范

引脚号	缩写符	信号方向	说明
1	PG	保护接地	公共端
2	TXD	输出	发送数据
3	RXD	输入	接收数据
4	RTS	输出	请示发送
5	CTS	输出	清除发送
6	DSR	输入	数据设备准备好
7	SG	公共端	信号地
8	DCD	输出	载波检测
15	RCLK	输入	接收时钟
17	RCLK	输入	接收时钟
18	TEST	输出	本地回环
20	DTR	输出	数据终端准备好
21	RLB	输出	远程回环
22	RI	输入	振铃指示器
24	DTECLK	输出	发送时钟（DTE）
25	TI	输入	测试指示器

表 4-8 V.36/RS-449 接口规范

引脚号	缩写符	信号方向	说明
管壳	PG	公共端	保护接地
4	TXD+	输出	发送数据
5	TRXC+	输入	发送时钟 (DCE)
6	RTD+	输入	接收数据
7	RTS+	输出	请求发送
8	RTXC+	输入	接收时钟 (DCE)
9	CTS+	输入	清除发送
10	TEST	输出	本地回环激活
11	DSR+	输入	数据设备就绪
12	DTR+	输出	数据终端就绪
13	DCD+	输入	数据载波检测
14	RLB	输出	远程回环
15	RI	输入	振铃指示器
17	CLK+	输出	发送时钟 (DCE)
18	TI	输入	测试指示器
19	GND	公共端	公共端返回
22	TXD-	输出	发送数据
23	TRXC-	输入	发送时钟 (DCE)
24	RXD-	输入	接收数据
25	RTS-	输出	请求发送
26	RTXC-	输入	接收时钟 (DCE)
27	CTS-	输入	清除发送
29	DSR-	输入	数据设备就绪
30	DTR-	输出	数据终端就绪
31	DCD-	输入	数据载波检测
35	CLK-	输出	发送时钟 (DTE)

表 4-9 EIA-530 接口规范

引脚号	缩写符	信号方向	说明
1	PG	公共端	保护接地
2	TXD+	输出	发送数据
3	RXD+	输入	接收数据
4	RTS+	输出	请求发送
5	CTS+	输入	清除发送
6	DSR+	输入	数据设备就绪
7	SG	公共端	信号地
8	DCD+	输入	数据载波检测

(续)

引脚号	缩写符	信号方向	说明
9	RTXC-	输入	接收时钟 (DCE)
10	DCD-	输入	数据载波检测
11	CLK-	输出	发送时钟 (DTE)
12	TRXC-	输入	发送时钟 (DCE)
13	CTS-	输出	清除发送
14	TXD-	输出	发送数据
15	TRXC+	输入	发送时钟 (DCE)
16	RXD-	输入	接收数据
17	RTXC+	输入	接收时钟 (DCE)
18	TEST	输出	本地回环
19	RTS-	输出	请求发送
20	DTR+	输出	数据终端就绪
21	RLB	输出	远程回环
22	DSR-	输入	数据设备就绪
23	DTR-	输出	数据终端就绪
24	CLK+	输出	发送时钟 (DTE)
25	TI	输入	测试指示器

第5章 数据链路层

要在一条通信线路上传送数据，除了必须建立一条物理线路（物理层的功能）之外，还必须有一些规程或协议来控制这些数据的传输，以保证被传输数据的正确性。实现这些规程或协议的硬件和软件加上物理线路就构成了本章要介绍的“数据链路层”（Data Link Layer, DLL）。

我们知道，物理层中也有许多规程或协议，但它们是用来构建物理传输线路、建立物理意义的网络通信，不是用来控制数据传输的。设计数据链路层的主要目的就是在原始的、有差错的物理传输线路的基础上，采取差错检测、差错控制与流量控制等方法，将有差错的物理线路改进成逻辑上无差错的数据链路，以便向它的上一层——网络层提供高质量的服务。就像我们修好了路，还得制定一些交通法规，使路上行驶的车辆必须按照一定的规则行驶，否则可能会经常出现交通事故。这些“交通法规”也为了使这些车辆到达某个车站（这里所说的“车站”就相当于计算机网络体系结构中的“网络层”）时能有序进、出站，最终使这条数据通信之“路”发挥它本来的作用。

在不同网络体系结构中，数据链路层的结构和所包括的功能并不完全一样。本章主要针对广域网中的数据链路层和局域网体系结构中的逻辑链路控制（LLC）子层的功能及相关技术进行全面、深入的介

绍。读者要着重掌握的是数据链路层的链路管理、数据帧封装、差错控制、流量控制这几项功能的实现原理，BSC、SDLC、HDLC和PPP这几种典型的数据链路层协议及二交换原理。有关局域网体系结构中的媒介访问控制（MAC）子层的功能和相关技术将在下章介绍。

5.1 数据链路层基础

在所有计算机网络体系结构中都直接或间接地包含了数据链路层（在TCP/IP协议体系结构中数据链路层的功能包含在网络访问层中）。数据链路层和它下面的物理层其实本质作用都是一样的，就是用来构建进行网络通信、访问的通道，只不过物理层构建的是一条物理通道，而数据链路层构建的是真正用于数据传输的逻辑通道。正因如此，在目前Internet中广泛使用的TCP/IP协议体系结构中，物理层和数据链路层是集中划分在网络访问层这一层之中的。

5.1.1 划分数据链路层的必要性

虽然说物理层和数据链路层的本质作用都是用来构建网络通信、访问通道，但它们所建立的通信通道是不一样的。首先要说明的一点是，在物理层上构建的是物理链路，在数据链路层上构建的是逻辑链路或者数据链路，它们是不同的概念，但确实有许多读者分不清楚。

物理链路是指在物理层设备（包括传输介质、物理接口和收发器等）和相应物理层通信规程作用下形成的物理线路，是永久存在的，且是不可删除的（除非物理拆除）；逻辑链路则是通信双方在需要进行数据通信时，在数据链路层设备和相应的通信规程作用下建立的逻辑链路，可以是永远存在的（如局域网中的以太网链路），也可以不是永久存在的（如广域网中的链路），是否永久存在要视具体的数据链路层服务类型而定。这里还有一个“链路”的概念，它是指相邻节点之间的那段数据线路。

在看到物理链路和逻辑链路之间区别的同时，又要看到它们之间的联系，那就是逻辑链路必须建立在物理链路之上。如果通信双方的物理线路都不通，是不可能建立用于数据传输的逻辑链路的。我们可以这样来理解它们之间的关系，物理链路是基础线路，相当于一条公路的路基，而逻辑链路是在物理链路之上的高级线路，可以理解为在公路上铺设了柏油或者水泥的车道。它们之间的关系如图5-1所示。

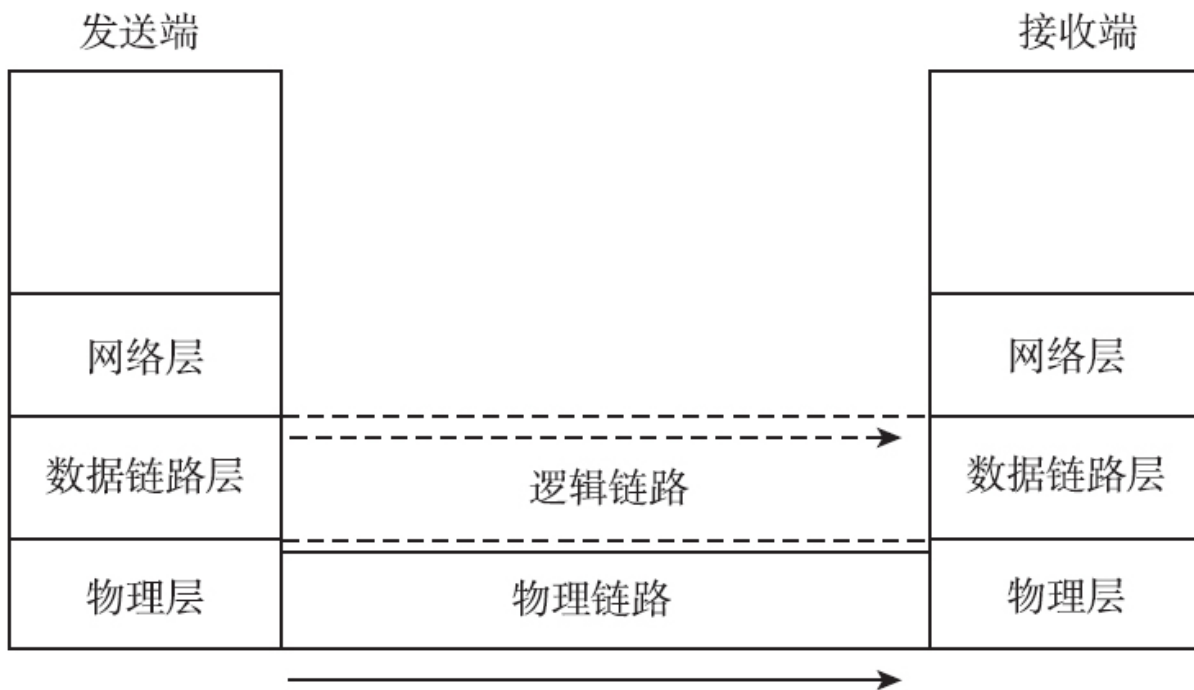


图 5-1 “物理链路”和“逻辑链路”的关系

无论在多么复杂的网络中，从逻辑意义上来讲，真正的数据传输通道就是数据链路层中所定义的数据链路，只不过在要经过多个网络的数据通信中，数据链路是分段的，每个网络都有一段链路，这些链路段连接起来就是整个数据通信的数据链路。图5-2所示的是一个有三个网段经过两个路由器（路由器A和路由器B）相连的网络，现假设PC用户A要向PC用户B发送一个数据，它的实际数据传输过程如图5-2中各网络物理层之间的实线箭头所示，而逻辑上可以等同各部分数据链路层之间构建的“逻辑链路”之间的数据转发，如图5-2中虚线箭头所示。

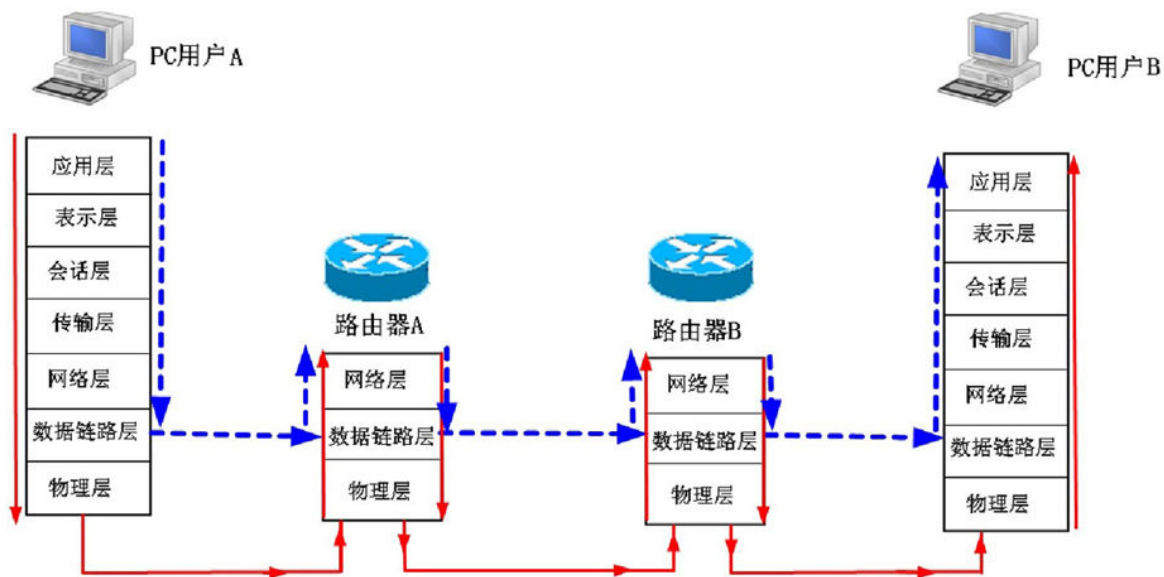


图 5-2 数据在网络中的传输方向

说到这里，可能有的读者会问，既然在“物理层”中构建了数据传输通道（也就是物理层中的“信道”），那为什么还要多加一个“数据链路层”的功能呢？其实这主要有两方面的原因：

□一是由于物理层传输介质的多样性，通信规程也各不相同，性能不稳定，而数据链路层中构建的逻辑链路不考虑不同物理链路上传输介质及其通信规程上的区别（也就是我们通常所说的数据链路层可以屏蔽物理层中传输介质的不同），只是从逻辑意义上构建一条性能稳定、不受传输介质类型影响的逻辑数据传输通道。就像修建公路时，所用的材质也可能不一样，有的用普通的泥巴，有的用沙石，还有的用大石材。如果仅靠一些基础材料修建公路，可能修好的公路通车性能很差，有的甚至根本不能通车，只能步行。但如果我们在这些

公路上再统一铺一层钢筋混凝土，那么这些由不同材料修建的公路就可能满足基本相同的通车性能的要求了。

□再一个原因是，在物理层中数据是一位位地单独传输的，不仅数据传输效率低下，而且容易出现数据传输差错（如出现某些数据位丢失或者错位，在物理层中又没有相应的通信规程进行数据传输差错控制），就像一条不能通车的普通公路上，人只能一个个地步行，还可能出现迷路现象一样。而在数据链路层中数据是以“帧”为单位进行传输的，一个帧通常是有数千个比特位的，不仅传输效率提高，还不容易出错（因为在数据链路层中有专门的通信规程来负责数据传输差错控制），就像在能通车的公路上以车为单位运载人一样，不仅传输效率提高，还不容易出现各种交通事故。至于数据链路层的主要功能，将在本章后具体介绍。

5.1.2 数据链路层结构

在正式介绍“数据链路层”主要功能和实现原理前，我们先要明白，各种计算机网络体系结构中，数据链路层的结构是不完全一样的。在OSI/RM和TCP/IP体系结构中，数据链路层就一层，而在局域网体系结构中是可细分为两个子层的，那就是逻辑链路控制（Logical Link Control, LLC）子层和介质访问控制（Medium Access Control, MAC）子层，如图5-3所示。

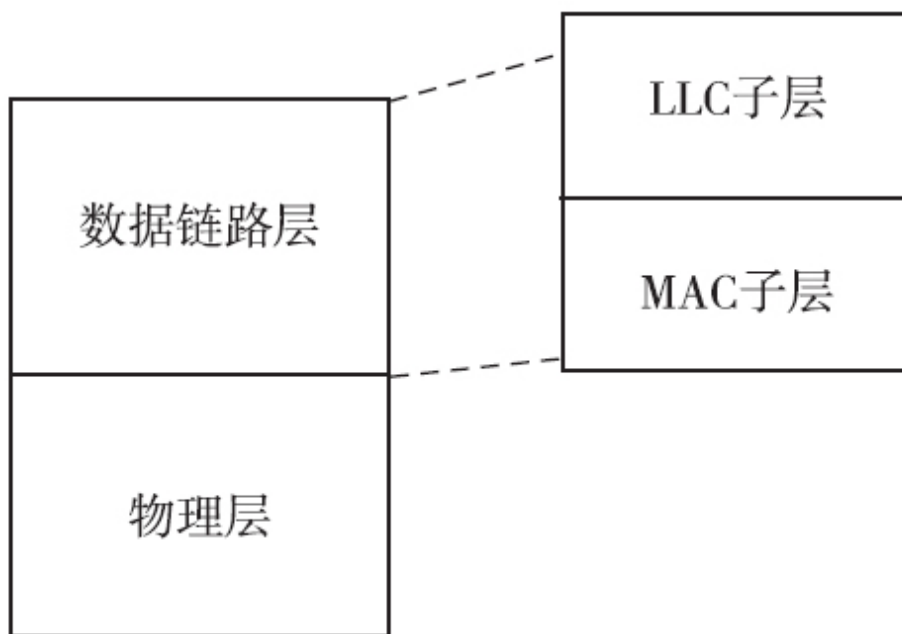


图 5-3 局域网体系结构“数据链路层”的两个子层

经验之谈 设置MAC子层的目的是主要是解决多路通信的介质争用和寻址，所以其主要适用于广播型链路和点对多点链路。对于点对点

链路来说，没什么太大意义，因为点对点链路不存在介质争用，是一路通信所独占的。具体将在下章介绍。

从图5-3可以看出，在数据链路层中，与它的下一层——物理层相邻的是MAC子层，与它的上一层——网络层相邻的是LLC子层。所以MAC子层接受物理层的服务，为LLC子层服务，而LLC子层则是接受MAC子层服务，为网络层服务。而各层（其他层也一样）之间接受服务或者提供服务的地方就是SAP（Service Access Point，服务访问点）。下面先来了解什么是SAP。

1.各层的SAP

从SAP的中文名称“服务访问点”可以看出，它就是上层访问相邻下层所提供服务的点。我们知道，在计算机体系结构中，下层是为相邻的上层提供服务的，而下层对它的所有上层都是透明的。也就是上层不会具体管它的下面各层是如何工作的，只需要它的相邻下层提供必要的服务即可。

SAP是邻层实体（“实体”也就是对应层的逻辑功能）间实现相互通信的逻辑接口，位于两层边界处。从物理层开始，每一层都向上层提供服务访问点（应用层除外），每一层都有SAP，但不同层的SAP内容和表示形式都是不一样的。各层SAP的表示形式是对应层第一个单词的第一个字母加上“SAP”，如物理层SAP表示为PSAP（Physical layer

Service Access Point），对应的就是网络通信中设备的具体物理接口；数据链路层SAP表示为DLSAP（Data Link Control layer Service Access Point），对应的就是各个物理接口的MAC地址；网络层SAP表示为NSAP（Network layer Service Access Point），对应的就是各物理接口上配置的网络地址（如IP地址，但在OSI/RM中网络地址不一定是IP地址，要视具体的网络层协议而定，如还可以是IPX地址）；传输层SAP表示为TSAP（Transport layer Service Access Point），对应的就是具体网络应用通信所用的传输层端口；会话层SAP表示为SLSAP（Session layer Service Access Point），对应的就是具体网络应用会话进程；表示层SAP表示为PLSAP（Presentation layer Service Access Point），对应的就是具体网络应用进程中的用户标识。

从以上介绍可以得知，其实SAP每层所对应的“地址”，但是针对一个具体的网络通信（注意，这里特别说明一下不是数据通信）来说，不同层中的SAP数是不一样的。如物理层的PSAP只有一个（就是对应的物理接口），数据链路层的DLSAP也只有一个（就是对应物理接口的MAC地址），在网络层中虽然每个物理接口可以有多个IP地址，但是对于一个具体的数据通信来说，它也只有一个，所以NSAP也只有一个，传输层及以上各层的SAP就可以有多个了，因为每一个网络通信中可以同时进行多路网络应用（当前有多少个网络应用进程，就需要多少个SAP），实现多路数据通信。正因如此，针对一个具体的网络通信中各层的SAP可以描述为图5-4所示的形式。

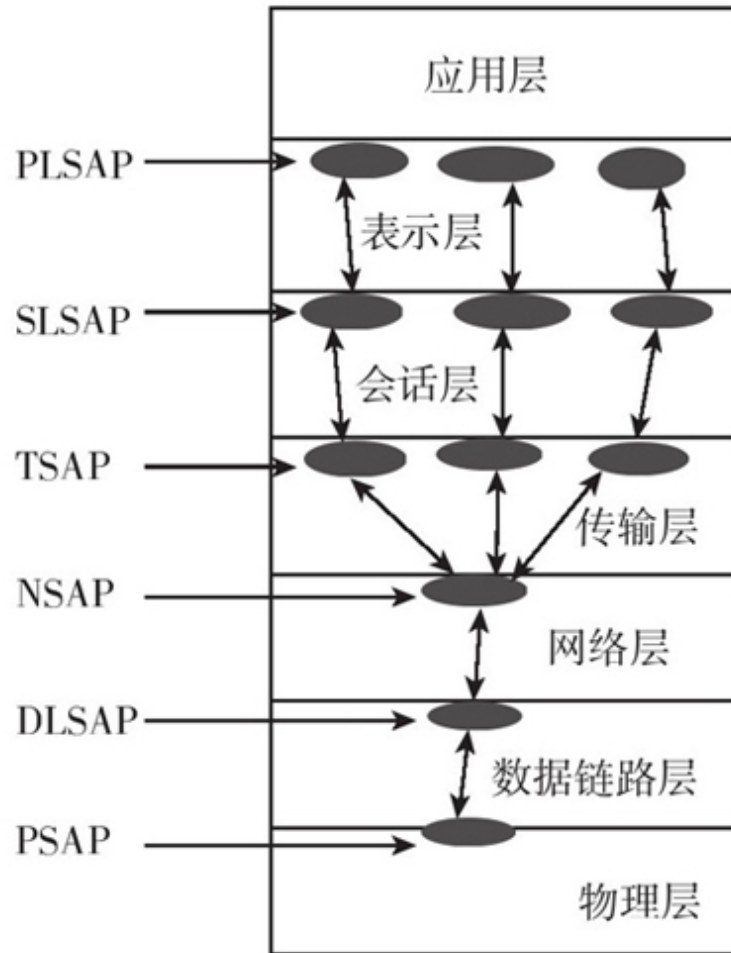


图 5-4 OSI/RM各层的SAP

2.MAC子层

从“MAC”的中文名称“介质访问控制”可以看出，MAC子层的最基本功能就是如何控制不同用户数据传输中对物理层传输介质的访问，其中包括介质访问时的寻址（这里是通过MAC地址进行的），以及解决可能发生的介质访问冲突（也就是我们通常听到的“仲裁介质的使用权”，即规定站点何时可以使用通信介质）。如IEEE 802.3以太网标准

MAC子层规范了如何在总线型网络结构下使用传输介质；IEEE 802.4令牌总线（Token-Bus）标准MAC子层规范了如何在总线的网络结构下利用令牌（token）控制传输介质的使用；IEEE 802.5令牌环（Token-Ring）标准MAC子层规范了如何在环状网络结构下利用令牌来控制传输介质的使用；IEEE 802.11标准的无线局域网标准MAC子层规范了如何在无线局域网的结构下控制传输介质的使用。

具体而言，数据链路层中与各种传输介质访问有关的问题都放在“MAC子层”来解决。其主要功能包括：数据帧的封装/卸装，帧的寻址和识别，帧的接收与发送，帧的差错控制、介质访问冲突控制等。有关MAC子层的具体功能和技术介绍将在下章进行。

3.LLC子层

从“LLC”的中文名称“逻辑链路控制”可以看出，LLC子层的最基本功能就是负责数据链路层中逻辑链路（逻辑链路就是物理层信道中的物理链路在通过LLC子层协议作用后形成的虚拟链路）的控制，其中包括逻辑链路的建立和释放，控制信号交换、数据流量控制，解释上层通信协议传来的命令并且产生响应，以及克服数据在传送的过程当中所可能发生的种种问题，如数据发生错误、重复收到相同的数据、接收数据的顺序与传送的顺序不一致等。在LLC子层方面，IEEE 802系列标准中只制定了一种标准——IEEE 802.2，各种不同局域网都使用相同的LLC子层通信标准。

由于网络层上可能有多种通信协议同时存在，而且每一种通信协议又可能同时与多个对象沟通，因此当LLC子层从MAC子层收到一个数据包时必须能够判断要送给网络层的是哪一个通信协议。为了达到这种功能，在LLC子层中提供了“数据链路层”的SAP，作为与“网络层”通信交互的接口（每路通信需要一个SAP接口，如图5-5所示）。为了能够辨认出LLC子层上传送的数据从哪里来，要到哪里去，在LLC子层上传输的每个LLC数据单元（LLC Protocol Data Unit, LPDU）上都会有“目的服务访问点”（Destination Service Access Point, DSAP）和“源服务访问点”（Source Service Access Point, SSAP）这两个地址。具体的LPDU帧格式将在本章后面介绍。

在计算机网络中进行的数据传输，虽然实际上是从发送端的高层一路经过数据链路层、物理层，然后再从接收端的物理层、数据链路层一直传输到对应的高层（如图5-5中实线箭头所示），而从逻辑意义看，数据是从发送端数据链路层到接收端数据链路层间的一段段逻辑链路上进行传输的（如图5-5中虚线箭头所示），因为在物理层中传输的比特流最终还是要转换成数据帧在数据链路层中传输。

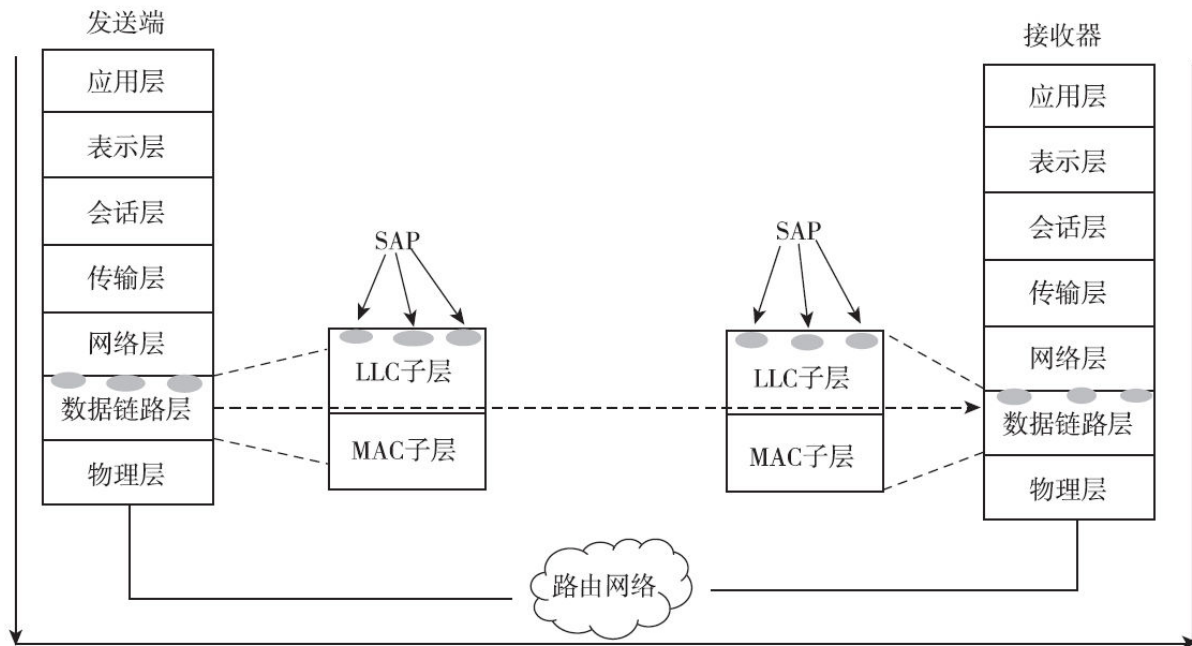


图 5-5 LLC子层的SAP及数据传输原理示例

5.2 数据链路层主要功能及实现原理

数据链路层位于网络体系结构中“网络层”（在TCP/IP协议体系结构中称网际互连层）的下层，所以它的一项基本功能就是向网络层提供透明、可靠的数据传输服务（在计算机网络体系结构中，下一层是为相邻的上一层服务的）。“透明”是指要使在数据链路层上所传输的数据在内容、格式及编码上都有限制，也就是要使一些本来用于特殊用途的控制字符（具体有哪些控制字符将在本章后面介绍）也能像正常的数据一样传输，使接收端不要误认为这些字符为控制字符；可靠的传输使数据从发送端无差错地在数据链路上传输到目的接收端。总体而言，数据链路层（其实这里主要是针对LLC子层）的主要功能就是四个方面：数据链路管理、封装成帧、透明传输、差错控制。下面具体介绍。

5.2.1 数据链路管理

在本章前面说到了，在数据链路层中要形成一条更有利于数据传输的数据链路，而不是直接利用下面物理层中建立的物理链路。物理链路在没有人拆除时是永久存在的，而数据链路一般是非永久存在的（但局域网中的数据链路是永久的），仅当有数据传输时建立并存

在，在数据传输完后自动拆除。数据链路是由数据链路层中的LLC子层通过相应的通信规程（也就是通常所说的协议）建立并管理的。

说明 数据链路分为点对点链路和点对多点链路（或“广播链路”）两种。点对点链路就是一个节点只与另一个节点连接起来的链路，用于建立点对点通信。它所采用的是点对点协议，如PPP（点对点协议），PPPoE（基于以太网的点对点协议）。点对多点链路就是一个节点同时与多个节点连接建立起来的链路，用于建立点对多点通信。它所采用的通常是点对多点协议，如以太网协议、WLAN协议，还有本章后面将要介绍的HDLC（高级数据链路控制）协议。

1.数据链路层提供的服务类型

根据数据链路层协议的不同，所建立的数据链路类型也会有不同。同时我们知道，数据链路层是为上面的网络层提供服务的，所以这些不同协议所建立的数据链路向网络层提供的服务类型也有所不同。总体上可把这些数据链路服务分为以下三类：有确认的面向连接服务、有确认的无连接服务、无确认的无连接服务。前者称为面向连接服务（Connection-oriented Service），后面两者称为无连接服务（Connetionless Service）。

说明 其实不仅本章中所讲的数据链路层协议有面向连接和无连接两种服务类型，在网络层和传输层协议中也有这两种类型的，如网

网络层中的X.25协议是面向连接的，而IP协议则是无连接的；传输层中的TCP协议是面向连接的，而UDP协议是无连接的。这些将在第10章进行介绍。

(1) 有确认的面向连接服务

有确认的面向连接服务里面包括两层含义：一是在提供服务时，必须先建立好双方通信连接；二是在提供服务时，必须要求对方确认后才进行。这种服务类型存在三个阶段，即数据链路建立、数据传输、数据链路释放等阶段。举个现实中的例子，就像我们打电话，我们打电话给某个人时，首先就是要拿起电话拨号（相当于建立连接的过程），然后对方拿起电话，问一下看是不是打错了（这就是一个“确认”过程）。确认不是打错的电话后，双方开始通话，这就相当于在数据链路中进行数据传输的过程；通话完毕，双方挂掉电话，相当于链路释放的过程。

从以上这个打电话的例子可以看出，数据链路层中有确认的面向连接服务是独占链路的，只有在当前数据传输完成，释放了链路后，其他用户才可能与同一个接收端进行数据传输。就像你打电话给你的朋友时，其他人再打电话给你朋友听到的是忙音，只有等你结束了与你朋友的通话后，其他人才可以打通你那朋友的电话。同样，从以上分析可以得出，有确认的面向连接服务非常可靠，这一则是因为有专门的通信链路，在一路通信使用某条链路时，其他通信不能同时使用

这条链路；再则是这种服务类型不会向错误的接收端进行数据传输，也可确认接收端正确地接收了发送来的数据，而且是按数据帧发送顺序接收，每一帧只接收了一次，因为它规定接收端在接收到每一个数据帧（每个帧都有编号）后必须对发送端进行确认，就像打电话一样，只有对方确认你是要找他的，他才可能接听你的电话。

大多数广域网中通信子网的数据链路层协议采用有确认的面向连接服务，如SLIP（串行线路协议）、PPP（点对点协议）、PPPoE（基于以太网的点对点协议）、HDLC（高级数据链路控制）协议等。

（2）有确认的无连接服务

有确认的无连接服务与有确认的面向连接服务的相同之处就是接收端在接收到的每一个数据帧时都向发送端确认；不同之处在于它在进行数据传输前是不需要建立专门的数据链路的，自然也不需要数据链路传输结束后释放数据链路（事实上是因为这类服务所用的数据链路已建立起来，而且是永久存在的，所以不用另外建立，如局域网中的链路）。就像我们从快递公司寄快递信件一样，信的投递路线我们不用管（事实上投递路线已经有了），但是在收件人收到信件时必须要求收件人签收（也就是要对接收到的每一个数据帧进行确认）。

有确认的无连接服务虽然不用建立专门的连接，但仍可以保证数据的可靠传输，因为它有“确认”功能，如令牌环网和令牌总线网中的

数据传输就是采用这种服务类型，接收端在接收到一个数据帧时会发送确认信息给发送端的。这类服务的另外一个主要用途就是用于一些不可靠信道中的数据传输，如各种无线通信系统。

(3) 无确认的无连接服务

无确认的无连接服务与前面的有确认的无连接服务的相同之处就在于它们都不需要在进行数据传输前先建立专门的数据链路，也就是无须先在通信双方建立通信连接；不同之处就是它在进行数据传输时不要求接收端对所接收到的每一个数据帧进行确认。就像我们从邮局寄平信一样，信件投递路线我们不用管（事实上投递路线已经有了），而且当信件到达收信人时，也不用收件人签名确认。

这种服务类型看似不可靠，但它是建立在可靠的通信线路基础之上的，所以数据传输仍然是非常可靠的。如我们常用的以太网中所使用的各种以太网协议就是采用这种服务的，因为以太网中的数据链路性能非常好，数据可靠传输有保障。在以太网中的数据链路始终是存在的，不用另外建立，在以太网中进行数据传输时接收端也不用对接收到的每一帧进行确认。

2.数据链路管理

LLC子层的链路管理功能主要是针对前面所介绍的有确认的面向连接服务类型（主要应用于广域网中）。它包括三个主要阶段：链路

建立、链路保持、链路释放。

在这种数据链路层服务中，链路两端的节点要进行通信前，发送端的数据链路层必须先确认对方已处于就绪状态，并交换一些必要的信息以对帧序号进行初始化，然后双方才能建立连接；在传输过程中为个数据连接是要持续保持的；如果出现差错，需要重新初始化，重新自动建立连接。传输完毕后则要释放所占用的数据连接，以供其他通信所用。

数据连路层的这种链路连接的建立、维持和释放过程就是数据链路层的链路管理功能。在多个站点共享同一物理信道的情况下（例如在局域网中），如何在要求通信的站点间分配和管理信道也属于数据链路层管理的范畴。

5.2.2 数据帧封装和透明传输

我们知道数据链路层位于物理层和网络层之间。在发送端，数据链路层是接收来自网络层的数据分组，而在接收端它是接收来自物理层的比特流，所以数据链路层的成帧功能就包含两方面的含义：一是将来自网络层的数据分组封装成数据帧，二是将来自物理层的一个个比特流组装成数据帧。因为帧封装（将物理层比特流组装成帧时，称为帧同步）通常是与透明传输一起考虑并实现的，所以在此一并介绍。不过本节仅介绍其基本原理，在本章后面还会结合具体的数据链路层协议详细介绍这些帧封装和透明传输原理。

1. 数据包的帧封装原理

通过前面的学习我们就已经知道，网络层传输的包（packet，又称分组），在数据链路层中传输的是“帧”（frame）。数据包到达数据链路层后加上数据链路层的协议头和协议尾就构成了一个数据帧。在每个帧的前部加上一个帧头部，在帧的结尾处加上一个帧尾部，把网络层的数据包作为帧的数据部分，就构成了一个完整帧。帧头和帧尾就是作为帧的起始和结束标志，也就是帧边界，如图5-6所示。

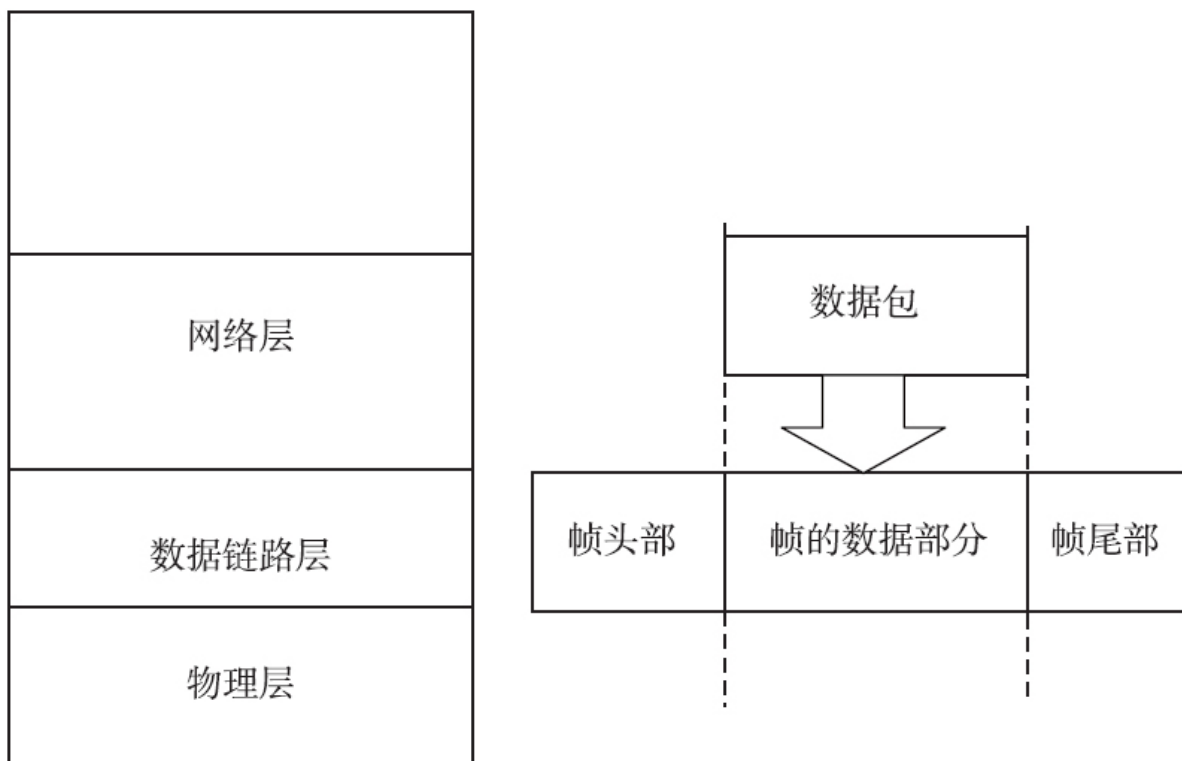


图 5-6 网络层数据包封装成帧的示意图

由数据包封装成的数据帧其大小是受对应的数据链路层协议的MTU（最大传输单元）限制的，如以太网数据链路层封装网络层IP包的MTU值为1500字节（这是指帧中数据部分，也就是来自网络层整个数据分组，最大不能超过1500字节，但不包括帧头和帧尾部分）。同时，帧还有最小大小限制，如以太网帧中封装的IP包最小值为46字节，如果封装的IP包小于最小帧要求时，就要用一些特殊字符进行填充，以满足对应链路中传输最小帧的限制。

2.比特流的帧组装及透明传输原理

在发送端数据链路层中的帧到达物理层后就会以比特位为单位进行传输，而不是以帧为单位进行传输。尽管在并行传输方式中，可以一次传输一个或多个字节，但每条线路中的传输单位还是比特位。发送端以比特位方式一位位地传输到接收端的物理层，然后接收端的物理层把比特流向数据链路层传输，到达后又要将比特流封装成数据帧，这就是数据链路层的帧组装方式了，其实也就是我们前面提到的帧同步问题。帧同步的目的就是要使接收端的数据链路层对从物理层传输而来的一串串比特流以帧为单位进行区分。

本节先简单介绍以下几种常用的帧同步方法的基本同步原理：字节计数法、字符填充的首尾定界符法、比特填充的首尾定界符法、违法编码法。本章的后面在介绍具体的数据链路层协议时还将具体介绍它们所采用的同步方法。

（1）字节计数法

这是一种以一个特殊字符代表一个帧的起始，并以一个专门的字段来标识当前帧内字节数的帧同步方法。接收端可以通过对该特殊字符的识别从比特流中区分出每个帧的起始，并从专门字段中获知每个帧后面跟随的“数据”（Data）字段的字节数，从而可确定出每个帧的结束位置。

这种面向字节计数的同步规程的典型实例是DEC公司的DDCMP（Digital Data Communications Message Protocol，数字数据通信报协议）。在DDCMP协议通信中，数据是在源站点与从站点之间以编号的数据消息的形式进行交换的，而从站点是以未编号的响应和控制消息的形式向主站点返回的。下面看看在这个协议的数据帧中如何实现帧同步，或者是如何成帧的：

在DDCMP协议帧格式（如图5-7所示）中，SOH字段是一个帧的帧头开始部分，有其固定的值（十进制为129，八进制值为201，十六进制值为81），就相当于一个帧开始的特殊字符；在NUM字段中为每个数据帧分配一个编号，从“1”开始，并以“1”为增量进行递增，最大值为256（也就是模为256），以确保在从站点中的正确消息序列，同时在COUNT字段中指出本数据帧中DATA字段的大小，这些都是用来进行帧同步的。

8	14	2	8	8	8	16	8 ~ 131064	16bits
SOH	COUNT	FLAGS	RESP	NUM	ADDR	BLKCK1	DATA	BLKCK2

图 5-7 DDCMP帧格式

(2) 字符填充的首尾定界符法

该同步方法是用一些特定的控制字符来定界一个帧的起始与结束，如IBM的BSC协议在每个数据块的头部用一个或多个同步字

符“SYN”来标记数据块的开始；尾部用字符“ETX”来标记数据块的结束。图5-8所示的是要传输一个“ADFGJ”的字符串，在帧的头部加上了两个SYN控制字符，用于标识该帧的开始，在结束位置加了ETX控制字符，用于标识该帧的结束。

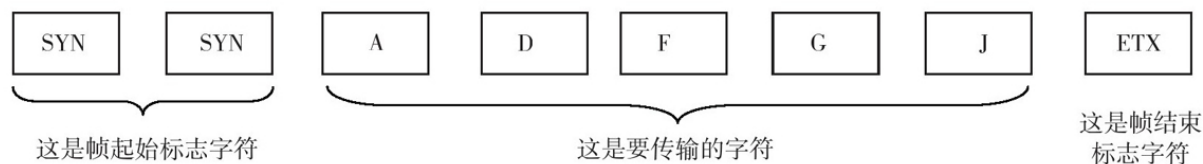


图 5-8 字符填充的首尾定界符帧同步访法应用示例

另外，为了不使数据信息中与以上特定字符相同的字符被误判为帧的首尾定界符，可以在这种数据帧的帧头填充一个转义控制字符（Data Link Escape-Start of TeXt，DLE），这就属于“透明传输”的范围了，这部分内容将在本章后面介绍具体的数据链路层协议时介绍。

（3）比特填充的首尾定界符法

该帧同步方法是通过在帧头和帧尾各插入一个特定的比特串（如01111110）来标识一个数据帧的起始与结束，这个帧头、帧尾特定比特串称为帧标志符。如传输的比特流为1001110101，组装成帧后就是0111111010011101010111110。

而为了透明传输，也就是为了避免在信息位中出现的与帧起始和结束标志符相似的比特串时被误判为帧的首、尾标志，采用了比特填充的方法。比如上面采用的特定模式为“01111110”，则对信息位中的任何连续出现的5个“1”（因为帧标志符中是有5个连续“1”），发送端自动在其后插入一个“0”。如要传输的数据帧为“0110111111011111001”（因为其中有5个连续的“1”，很可能被误认为是帧首、尾标志），采用比特填充后，实际传送的是

0111111001101111101011111000101111110（注意，前、后面两个“01111110”是帧首、尾标志符）。另外，因为在原信息中，有一段比特流与帧标志符类似，为了与用于标识帧头和帧频尾的特定模式字符区别，在有5个连续“1”的比特位后面加插入一个“0”（斜体“0”）。而接受方在收到上述最终数据后进行发送端的逆操作，首先去掉两端的特定模式字符，然后在每收到连续5个“1”的比特位后自动删去其后所跟的“0”，以此恢复原始信息。

比特填充帧同步方式很容易由硬件来实现，性能优于字符填充方式。所有面向比特的同步控制协议均采用统一的帧格式，不论是数据，还是单独的控制信息均以帧为单位传送，其典型代表是ISO的HDLC协议，在它的首尾均有标志字段（Flag，8位，即01111110），具体将在本章后面介绍。

（4）违法编码法

该帧同步方法是在物理层采用特定的比特编码方法时采用。例如，曼彻斯特编码方法，将数据“1”编码成“高-低”电平对（在半个码元处跳变，下同，具体参见4.4.2节），将数据“0”编码成“低-高”电平对。而高-高电平对和低-低电平对在数据比特中是违法的，因此可以借用这些违法编码序列来定界帧的起始与终止。违法编码法不需要任何填充技术，便能实现数据的透明性，但它只适用于采用冗余编码的特殊编码环境。

5.2.3 差错控制

说明 因数据链路层中的差错控制功能涉及比较多且比较复杂的技术，所以在此仅进行综合介绍，具体的差错控制技术将在本章后面介绍。

在上节介绍的“成帧”功能解决了帧同步问题，也就是接收端可以区分每个数据帧的起始和结束了，但是还没有解决数据正确传输的两方面问题：一是如果有帧出现了错误怎么办？二是如果有帧丢失了怎么办？这都是数据链路层确保向网络层提供可靠数据传输服务要解决的问题，也就是数据链路层的差错控制功能。

要实现差错控制功能，就必须具备两种能力：一具备发现差错的能力，二是具备纠正错误的能力。就像我们要发行一个“问题”小孩一样，你首先要知道他的“问题”在哪里，也就是存在问题的根源在哪里，然后才能采取适当的方法纠正这个“问题”小孩身上的问题。

1. 差错检测

在数据链路层检测数据传输错误的方法一般是通过对差错编码进行校验来实现，常见的校验方法有奇偶校验码（Parity Check Code，

PCC）、循环冗余校验（Cyclic Redundancy Check, CRC）两种。它们都统称为检错码（error-detecting code）。

奇偶校验码是一种校验代码传输正确性的方法，是根据被传输的一组二进制代码的数位中“1”的个数是奇数或偶数来进行校验的。采用“1”的奇数个校验的方法称为奇校验，反之称为偶校验，但采用何种校验是事先确定好的。具体做法是在传输的二进制代码最后专门设置一个奇偶校验位，用它使这组代码中1的个数为奇数或偶数（具体是偶数还是奇数，要视所采用的是偶校验还是奇校验），然后再在接收端进行校验，看里面的“1”的个数是否仍与原来一样的奇数或偶数，来确定数据传输的正确性。

循环冗余校验是一种根据传输或保存的数据而产生固定位数校验码的方法，主要用来检测或校验数据传输或者保存后可能出现的错误。生成的数字在传输或者储存之前计算出来并且附加到数据后面，然后接收端进行检验确定数据是否发生变化。

有关以上这两种差错检测方法在本章后面有专门的介绍。

2.差错纠正

在差错纠正方面，数据链路层针对不同的传输类型采取了不同的纠错方法。对于面向字符的异步传输（如键盘与主机的通信、ATM传输协议等）一般是采用“反馈检测”的方法来进行纠错。就是在接收端

收完一帧数据后，向发送端发送回所接收到的完整数据帧，由发送端通过与原始发送的帧进行比较来判断接收端是否正确接收了对应帧。如果判断是出了错，则发送端向接收端发送一个**DEL**字符及相应的帧信息，提示接收端删除对应的帧，然后重发该帧；否则表示接收端已正确接收了对应的帧，不重发对应的帧。但对于在传输过程中完全丢失的数据就不能采用这种纠错方法了，因为接收端根本没有收到这帧数据，所以也就不会向发送端发回反馈信息，发送端自然就不能确认接收端是否正确接收了对应的帧。

为了解决这个问题，通常在数据发送时引入计时器（**Timer**）来限定接收端发回反馈信息的时间间隔。当发送端发送一帧数据的同时启动计时器，若在限定时间间隔内没有收到接收端的反馈信息，即计时器超时（**Timeout**），则可认为传的对对应帧已出错，或丢失，继而发送端知道要重新发送对应的数据帧。同时，为了避免同一帧数据可能被多次重复发送，引发接收端多次收到同一帧并将其递交给网络层的危险，采用对发送的帧进行编号的方法，即同一个帧的编号是一样的，从而使接收端能从帧编号来区分是新发送来的帧，还是已经接收但又重新发送来的帧，以此来确定要不要将重新接收到的帧递交给网络层。

由于在“反馈检测法”中一帧数据会在信道中至少往返传输两次，传输效率低，所以一般不采用这种差错控制方法，而是采用一种称

为“自动重发请求”（ARQ法）的方法。实现原理就是先让发送端将要发送的数据帧附加一定的冗余检错码（如前面介绍的PCC、CRC码）一起发送，接收端则根据检错码对数据帧进行错误检测，若发现错误，就返回请求重发的响应（不用返回全部的帧），发送端收到请求重发的响应后，便重新传送该数据帧。

另外，还有一些编码本身具有自动纠正错误的能力，称为“纠错码”（Error-correcting Code），在数据链路层中常用的如海明码（Hamming Code）。

以上差错控制方法都将在本章后面具体介绍。

5.2.4 流量控制

“流量控制”包括两方面的含义：一是发送端的数据发送速度与接收端的数据接收速度要匹配，否则接收端来不及接收就会造成数据在传输过程中的丢失。这个很好理解，比如几个人站成一排传递货物时，如果中间有个人速度比较慢，而上面的人不断传来货物，肯定就会把来不及接收的货物放在地上（不进入正常的传递之中）。二是发送端的数据发送速度要与线路上的承载速率（与线路信道带宽有关）相匹配，否则也会造成数据在传输过程中的丢失。这个也很好理解，就像一条小河，上游来的水量很大，超过了小河所能承载的能力，这时上游来的水肯定就不会有原来那么大的流速（形成速率瓶颈），甚至可能会漫过小河河堤而流到外面。

注意 流量控制并不是数据链路层所特有的功能，许多高层协议中也提供流量控功能，只不过流量控制的对象不同而已。比如，对于数据链路层来说控制的是相邻两节点之间数据链路上的流量，而对于传输层来说控制的是从源到最终目的端之间的流量。

在数据通信中，由于通信双方各自使用的设备工作速率和缓冲存储的空间的差异，可能出现发送端发送能力大于接收端接收能力的现象，如若此时不对发送端的发送速率作适当的限制，在接收端前面来

不及接收的帧将被后面不断发送来的帧所“淹没”，从而造成帧的丢失。由此可见，数据链路层上的“流量控制”功能实际上是对发送端数据传输速率的控制，使其数据发送速率不超过接收端所能承受的数据接收能力。考虑到在接收端还需要对来自物理层的比特流进行一系列的处理，如帧封装，向发送端发送返回确认帧等，所以通常是要使发送端的发送速率略小于接收端的数据处理能力。

在数据链路层中进行流量控制的方案有两种：一是基于反馈的流量控制方案，二是基于速率的流量控制方案。

1.基于反馈的流量控制方案

“基于反馈的流量控制方案”就是接收端在接收到一个数据帧后，要向发送端发送一个确认帧，表示发送端可以继续向它发送数据了。这种方案也称“停-等”方案（也就是将在5.3.4节介绍的“空闲重发请求”方案），就是发送端在发送一帧数据后必须等待接收端返回确认响应消息，然后才能发送下一数据帧。接收端是通过检查帧的校验序列（FCS），无错则发送确认帧，否则不发送返回消息，表示该帧已出错，要求重发。

其实这也就是利用上节介绍的差错控制方案。但这里存在一个问题，就是在数据帧或确认帧丢失时，双方会无休止等待。解决这一问题的方法是发送后使用定时器，在发送端发送一帧数据时会启动这个

定时器，要求接收端必须在这个时间内返回确认帧，否则就认为这个数据帧已丢失，重发该数据帧。当然，与上节介绍的差错控制方案时提到的一样，这也可能造成接收端接收到多个一样的数据帧，解决这一问题的方法就是给每个帧加上一个编号，这样编号一样的帧接收端只需接收其中一个即可，其他的丢弃。

由此可以看出，“停-等”方案其实就是直接利用了上节介绍的纠错方案，这里提到的几种措施可以说是一环扣一环，最终可以既确保数据的可靠传输（也就是“差错控制”功能），又有效地控制了双方通信的流量（也就是“流量控制”功能）。

2.基于速率的流量控制方案

“基于速率的流量控制方案”是基于窗口滑动机制的速率控制方案，它规定发送端一次可以发送多少个数据帧，限制了发送端的数据传输速率，而无须接收端发回确认帧。这种机制等在本章后面也有专门介绍。其实它与第10章要介绍的传输层流量控制方案是极其类似的，不同的只是数据链路层的流量控制是针对链路两端的点对点控制，而传输层则是直接针对通信双方的端到端系统。

5.3 差错控制方案

前面已说到，数据链路层的差错控制功能分为差错检测功能和差错纠正功能。在差错检测方面通常在数据帧中加上一些具有检错功能的冗余代码（称为检错码，**Error-detecting Code**），然后根据这些冗余代码所表达的信息，以及接收端对数据帧的奇偶性重新计算的结果就可以发现数据帧在传输过程中是否出现差错。如前面说到的**PCC**（奇偶校验码）、**CRC**（循环冗余校验）都属于检错码。但它们均不具有自动纠错功能，不能确定是哪一个或哪一些位出错了，也不能纠正这些差错，通常也是与纠错方案中的自动请求重发法（**ARQ**法）结合起来进行差错控制的。

在差错纠正方面主要有反馈检测法和自动重发请求法两种（自动重发请求法中又分为几种，如空闲重发请求法，连续重发请求法等）。也有一些具有自动纠错功能的编码，称为纠错码（**Error-correcting Code**），又称前向纠错码（**Forward Error Correction**），如海明码（**Hamming Code**）。这些方案都将在下面各小节中具体介绍。

5.3.1 奇偶校验码检错方案

奇偶校验码（PCC）是奇校验码和偶校验码的统称，是一种有效检测单个错误的检错方法。它的基本校验思想是在原信息代码的最后添加一位用于奇校验或偶校验的代码，这样最终的帧代码是由 $n-1$ 位信元码和1位校验码组成，可以表示成为 $(n, n-1)$ 。加上校验码的最终目的就是要让传输的帧中“1”的个数固定为奇数（采用奇校验时）或偶数（采用偶校验时），然后通过接收端对接收到的帧中“1”的个数的实际计算结果与所选定的校验方式进行比较，就可以得出对应帧数据在传输过程中是否出错了。如果是奇校验码，在附加上一个校验码以后，码长为 n 的码中“1”的个数为奇数；如果是偶校验码，则在附加上一个校验码以后，码长为 n 的码中“1”的个数为偶数（0个“1”也看成是偶数个“1”）。奇偶校验方法可以通过电路来实现，也可以通过软件来实现，在此我们只要知道它校验的原理即可。

下面打个很简单的比喻来说明PCC的校验原理。现假设一学校要组织各班学生参加一次长跑运动，同时有几名外地学生想参加长跑运动，需在每班插入一个外地学生，但插入的学生性别不能随意，最终要求插入外地学生后各班的女学生数必须为偶数（相当于采用偶校验方式，当然也可以要求必须为奇数，若为奇数，则此时为奇校验方式），这样就可确定每班插入的这位外地学生的性别了。如果长跑后各班清点人数时，发现有些班的女生数量不是原来规定的偶数（或奇数），则证明这些班的学生在长跑过程中走乱了（相当于数据在传输

途中出现了差错) 或者有学生掉队 (相当于数据在传输途中丢失了) 了。这就是我们这里所说的奇偶校验原理。

我们知道ASCII字符是8位编码, 其中高7位是信息代码, 最后1位就是奇偶校验位。如现在传输的是ASCII字符, 如果采用奇校验, 则每个ASCII字符代码中“1”的个数均必须为奇数个, 如果发现某个字符中“1”的个数是偶数, 则这个ASCII字符在传输过程中肯定出错了; 同理, 如果采用的是偶校验, 则每个ASCII字符代码中“1”的个数均必须为偶数个, 如果发现某个字符中“1”的个数是奇数, 则这个ASCII字符在传输过程中也肯定出错了。其他采用奇偶校验方式的数据的校验原理也是一样的。

假设现在要传输一个ASCII字符, 它的高7位代码为1011010, 现在要采用奇校验方法, 则该字符的校验码为“1”, 整个ASCII字符代码就是1011010 1, 因为该字符中高7位信息代码中的“1”的个数是偶数个 (4个), 必须再加一个“1”才能为奇数; 同理, 如要采用偶校验方法, 则该字符的校验码为“0”, 整个ASCII字符代码就是1011010 0, 因为该字符中高7位信息代码中的“1”的个数已是偶数个 (4个), 所以最后一位中不能再是“1”, 只能为“0”。

这里要注意, 无论是采用奇校验, 还是偶校验, 每一信息的校验位是“0”还是“1”都是不固定的, 要视信息前 $n-1$ 位中“1”的具体个数而定。如果采用的是“奇校验”, 则要求所传输的 n 位信息中“1”的总个数

必须为奇数：如果前面 $n-1$ 位中“1”的个数已是奇数，则第 n 位的校验码位一定是“0”，只有这样才能确保整个 n 位信息中“1”的个数仍为奇数；如果前面 $n-1$ 位中“1”的个数是偶数，则第 n 位的校验码位一定是“1”，只有这样才能确保整个 n 位信息中“1”的个数为奇数。

如果采用的是偶校验，则要求所传输的 n 位信息中“1”的总个数必须为偶数（0个也看成是偶数个）：如果前面 $n-1$ 位中“1”的个数是奇数，则第 n 位的校验码位一定是“1”，只有这样才能确保整个 n 位信息中“1”的个数为偶数；如果前面 $n-1$ 位中“1”的个数是偶数，则第 n 位的校验码位一定是“0”，只有这样才能确保整个 n 位信息中“1”的个数仍为奇数。

另一个要注意的是，奇偶校验方法只可以用来检查单个码元错误，检错能力较差，所以一般只用于本身误码率较低的环境，如用于以太网中、用于磁盘的数据存储中等。如现在采用的是奇校验方法，传输的整个8位ASCII字符代码为10100100（最低位为校验位），如果里面前7位信息代码有一位出现了差错，由原来的“0”变为“1”，或者由原来的“1”变为“0”，都会改变里面“1”的个数，最终导致与对应的奇校验方法不一致，在这种情况下PCC能判断这个字符传输出错。但是如果里面前7位信息代码同时有2位或多位出现了差错，最终的结果可能会使字符的整个8位代码中“1”的个数奇偶性不变，而不能判断这个字符传输出错。如以上字符代码变为10001100，结果“1”的个数仍为

奇数个（3个），但实际上该数据已不是原来的数据了。这就为什么奇偶校验方法只能检测出单个码元错误的原因。

作为课后练习，大家自己再分析一下，如果所传输的二进制序列是11101100101，现要采用奇校验，则校验位的值是什么？采用偶校验呢？

5.3.2 循环冗余校验检错方案

上节介绍的奇偶校验码（PCC）只能校验一位错误，本节要介绍的循环冗余校验码（CRC）的检错能力更强，可以检出多位错误。

1.CRC校验原理

CRC校验原理看起来比较复杂、难懂，因为大多数书上基本上都是以二进制的多项式形式来说明的。其实其原理很简单，根本思想就是先在要发送的帧后面附加一个数（这个数就是用来校验的校验码，但要注意，这里的数也是二进制序列的，下同），生成一个新帧发送给接收端。当然，这个附加的数不是随意的，它要使所生成的新帧能与发送端和接收端共同选定的某个特定数整除（注意，这里不是直接采用二进制除法，而是采用一种称为模2除法的方法）。到达接收端后，再把接收到的新帧除以（同样采用模2除法）这个选定的除数。因为在发送端发送数据帧之前就已附加了一个数，做了去余处理（也就是已经能整除了），所以结果应该没有余数。如果有余数，则表明该帧在传输过程中出现了差错。

说明 模2除法与算术除法类似，但它既不向上位借位，也不比较除数和被除数的相同位数值的大小，只以相同位数进行相除。模2加法运算为： $1+1=0$ ， $0+1=1$ ， $0+0=0$ ，无进位，也无借位。模2减法运算

为：1-1=0，0-1=1，1-0=1，0-0=0，也无进位，无借位，相当于二进制中的逻辑异或运算。也就是比较后，两者对应位相同则结果为“0”，不同则结果为“1”。如100101除以1110，结果得到商为11，余数为1，如图5-9a图所示。再如11×11=101，如图5-9b图所示。

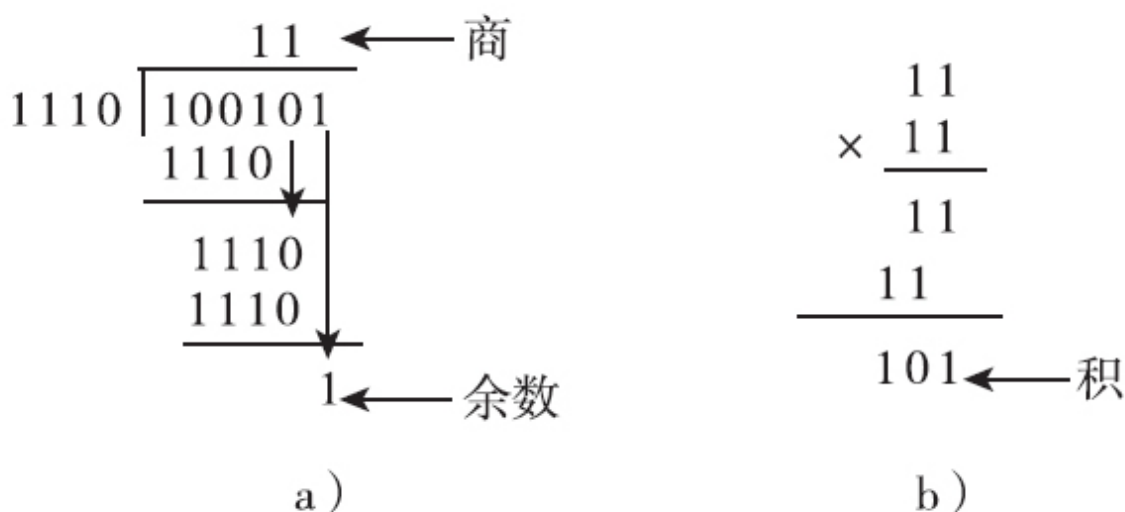


图 5-9 “模2除法”和“模2乘法”示例

具体来说，CRC校验的实现分为以下几个步骤：

1) 先选择（可以随机选择，也可按标准选择，具体在后面介绍）一个用于在接收端进行校验时，对接收的帧进行除法运算的除数（是二进制比特串，通常是以多项式方式表示，所以CRC又称多项式编码方法，这个多项式又称生成多项式）。

2) 看所选定的除数二进制位数（假设为k位），然后在要发送的数据帧（假设为m位）后面加上k-1位“0”，接着以这个加了k-1个“0”的

新帧（一共是 $m+k-1$ 位）以“模2除法”方式除以上面这个除数，所得到的余数（也是二进制的比特串）就是该帧的CRC校验码，又称FCS（帧校验序列）。但要注意的是，余数的位数比除数位数只能少一位，哪怕前面位是0，甚至是全为0（附带好整除时）也都不能省略。

3) 再把这个校验码附加在原数据帧（就是 m 位的帧，注意不是在后面形成的 $m+k-1$ 位的帧）后面，构建一个新帧发送到接收端，最后在接收端再把这个新帧以“模2除法”方式除以前面选择的除数，如果没有余数，则表明该帧在传输过程中没出错，否则出现了差错。

通过以上介绍，大家一定可以理解CRC校验的原理了。

从上面可以看出，CRC校验中有两个关键点：一是要预先确定一个发送端和接收端都用来作为除数的二进制比特串（或多项式）；二是把原始帧与上面选定的除进行二进制除法运算，计算出FCS。前者可以随机选择，也可按国际上通行的标准选择，但最高位和最低位必须均为“1”，如在IBM的SDLC（同步数据链路控制）规程中使用CRC——16（也就是这个除数一共是17位）生成多项式 $g(x)=x^{16}+x^{15}+x^2+1$ （对应二进制比特串为110000000000000101）；而在ISO HDLC（高级数据链路控制）规程、ITU的SDLC、X.25、V.34、V.41、V.42等中使用CCITT——16生成多项式 $g(x)=x^{16}+x^{15}+x^5+1$ （对应二进制比特串为110000000000100001）。

2.CRC校验码的计算示例

由以上分析可知，除数是随机或者按标准选定的，所以CRC校验的关键是如何求出余数，也就是校验码（CRC校验码）。

下面以一个例子来具体说明整个过程。现假设选择的CRC生成多项式为 $G(X)=X^4 + X^3 + 1$ ，要求出二进制序列10110011的CRC校验码。下面是具体的计算过程：

1) 首先把生成多项式转换成二进制数，由 $G(X)=X^4 + X^3 + 1$ 可以知道，它一共是5位（总位数等于最高位的幂次加1，即 $4+1=5$ ），然后根据多项式各项的含义（多项式只列出二进制值为1的位，也就是这个二进制的第4位、第3位、第0位的二进制均为1，其他位为0）很快就可得到它的二进制比特串为11001。

2) 因为生成多项式的位数为5，根据前面的介绍得知，CRC校验码的位数为4（校验码的位数比生成多项式的位数少1）。因为原数据帧10110011，在它后面再加4个0，得到101100110000，然后把这个数以“模2除法”方式除以生成多项式，得到的结果为0100，如图5-10所示。注意参考前面介绍的“模2除法”运算法则。

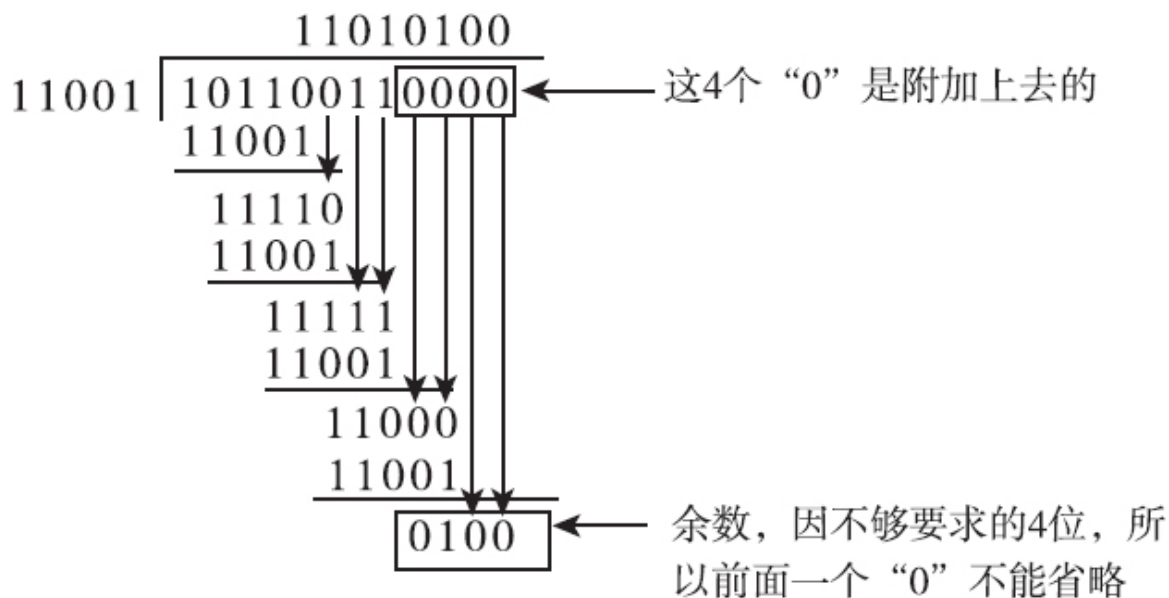


图 5-10 CRC校验码计算示例

3) 用上步计算得到的CRC校验替换帧101100110000后面的4个“0”，得到新的帧101100110100。再把这个新帧发送到接收端。

4) 当以上新帧到达接收端后，接收端会把这个新帧再用上面选定的除数11001以“模2除法”方式去除，验证余数是否为0，如果为0，则证明该帧数据在传输过程中没有出现差错，否则出现了差错。

通过以上对CRC校验原理的剖析和CRC校验码计算示例的介绍，大家应该对这种看似很复杂的CRC校验原理和计算方法比较清楚了。

下面大家做一个练习：假设CRC生成多项式为 $G(X)=X^5+X^4+X+1$ ，要发送的二进制序列为100101110，求CRC校验码是多少？

5.3.3 反馈检测法

“反馈检测法”是一种最简单、最容易实现，但并不常用的差错控制方法。这种差错控制方法的信道利用率低，因为在这种差错控制方法中每个数据帧均要求至少在信道中往返传输两次。下面具体介绍。

1.反馈检测法基本原理

反馈检测法不需要利用前面介绍的PCC（奇偶校验码）和CRC（循环冗余校验码）检测错码技术，其把差错检测和差错纠正的任务全部交给发送端。它要求接收端在接收到每一个数据帧后均要向发送端发送一个表示是否接收了该数据帧的反馈信息，且这个反馈信息就是原来由发送端发给接收端的原始数据帧。发送端在收到接收端发送的反馈信息后，通过对比保存在缓存中原来该帧的数据来判断接收端是否正确接收了该数据帧。

这种要求接收端必须向发送端反馈的做法，有些类似于我们向某人发送一个需要对方确认的重要邮件一样，如果收件人收到了你发送的这封邮件，要求对方对你进行回复确认；如果对方收到的邮件已遭破坏，也可以向你反映，然后你可以重发这封邮件给他，直到对方收到了正确的邮件。但也不完全与上述过程等同，因为这里的邮件反馈

并不需要发回原来邮件的全部内容，只是一条表示收到邮件了的反馈信息。

如果经过比较发现，接收端反馈来的某数据帧与原始帧不一样，则表明对应帧在传输过程中出现了差错，接收端接收到的对应帧是错误的，则接收端向发送端发送一个DEL字符和相应的帧编号（下面将介绍到），通知接收端删除对应的帧，由发送端重发对应的数据帧，直到接收到接收端反馈来的数据对应帧与原来发送的该帧数据完全一样为止；如果经比较接收端返回来的某帧与原来发送的对应帧完全一样，则表示接收端已正确接收到该数据帧，不再重发，继续发送下一数据帧。

2.两项附加技术

从以上反馈检测原理可以发现，这里有一个问题，那就是可能因一些原因导致一些数据帧在传输过程中完全丢失了，接收端根本没收到这些数据帧，自然也就不会向发送端发送对应数据帧的反馈信息了。就像你要求收件人在收到你发送的邮件时给你回信，但对方根本没收到你的邮件，自然也就不会有回信了，你还总在那里等他的回信。

为了避免出现这种情况，通常引入计时器（Timer）来限定接收端发回反馈消息的时间，即当发送端发送一帧数据时即自动启动计时

器，如果在计时器中限定的时间内仍没有收到接收端发来的反馈信息，则可认为该帧的数据传输出现了差错或丢失，就主动重新发送该帧。就像你对某人发送邮件前就告诉对方，“我现在就发送邮件给你，你必须在2小时内回复我是否收到了这封邮件，否则我认为你没收到这封邮件，会重新发一封给你”。

以上计时器方案虽然解决了由于数据丢失而引发的意外，但可能由于网络线路比较忙，送达的某帧数据在到达接收端时有所延迟，超出了计时器限定的时间范围，发送端仍然会重发对应的数据帧，这样就可能使接收端多次接收同一数据帧了。为了防止发生这种现象，在“数据链路层”又采取了“帧编号”（也就是“帧序列号”）方法对发送的每个数据帧进行编号（同一数据帧编号一样，无论发送多少次），从而使接收端能从该编号来区分是新发来的帧，还是已经接受但又重新发来的帧，从而确定要不要将接收到的帧递交给网络层。“数据链路层”通过使用“计数器”和“帧编号”两种措施来保证每个数据帧最终都能被正确地递交给目标网络层一次。

5.3.4 空闲重发请求方案

从上节介绍的“反馈检测法”差错控制原理中可以看出，其虽然简单，容易实现，也有较高的可靠性，但每个数据帧实际上在信道中均被传输了两次，造成信道利用率降低。也正如此，这种差错控制方法一般用于面向字符的异步传输中，因为在这种情形下，传输的数据量比较小，信道传输效率并不是主要问题。

实用的差错控制方法，既要求传输可靠性高，又要求信道利用率高。为此可使发送端将要发送的数据帧附加一定的冗余检错码（如PPC、CRC等）一并发送，接收端则根据检错码对数据帧进行差错检测；若发现错误，就返回请求重发的响应，发送端收到请求重发的响应后，便重新传送该数据帧。这种差错控制方法就称为自动重发请求（Automatic Repeat reQuest, ARQ）法。

ARQ差错控制方案仅需返回少量控制信息（接收端不必重传整个数据帧），便可有效地确认所发数据帧是否正确被接收。ARQ法有几种具体的实现方案，空闲重发请求（Idle RQ）和“连续重发请求”（Continuous RQ）是其中最基本的两种方案。本节先介绍“空闲重发请求”差错控制方案，下节将介绍连续重发请求差错控制方案。

空闲重发请求方案又称停-等（Stop and Wait）法，该方案规定发送端每发送一帧后就要停下来，然后等待接收端发来的确认信息（这就是停-等的意思），仅当接收端确认（ACK）信息后才继续发送下一数据帧。如果收到的是否认（NAK）消息，表示接收端接收的数据有错，请求发送端重发。另外，在计时器超时，发送端也会重发对应的帧。

图5-11a、b所示分别是正确接收数据时的数据帧发送流程和接收到有错误数据时的数据帧发送流程。

空闲重发请求差错控制方案的具体实现过程如下：

- 1) 发送端每次仅将当前数据帧作为待确认帧保留在缓冲存储器中，当发送端开始发送数据帧（如图5-11所示的data0）时，随即启动计时器。
- 2) 当接收端收到这个数据帧时，先利用帧中附带的检错码进行校验，确认无差错后，即向发送端返回一个确认信息（如图5-11a所示的ACK0、ACK1和图5-11b所示的ACK0）；当检测到该帧有错误时，向发送端返回一个否认帧（如图5-11b所示的NAK0），同时丢弃该帧。
- 3) 如果发送端在计时器中规定的时间内收到来自接收端的确认信息，即将计时器清零，清除缓存中的待确认帧，然后才开始下一数据帧（如图5-11a所示的data1）的发送；若发送端在规定时间内未收到来

自接收端的确认信息（即计时器超时），则重发存放于缓冲器中的待确认数据帧（如图5-11b所示的data1）。

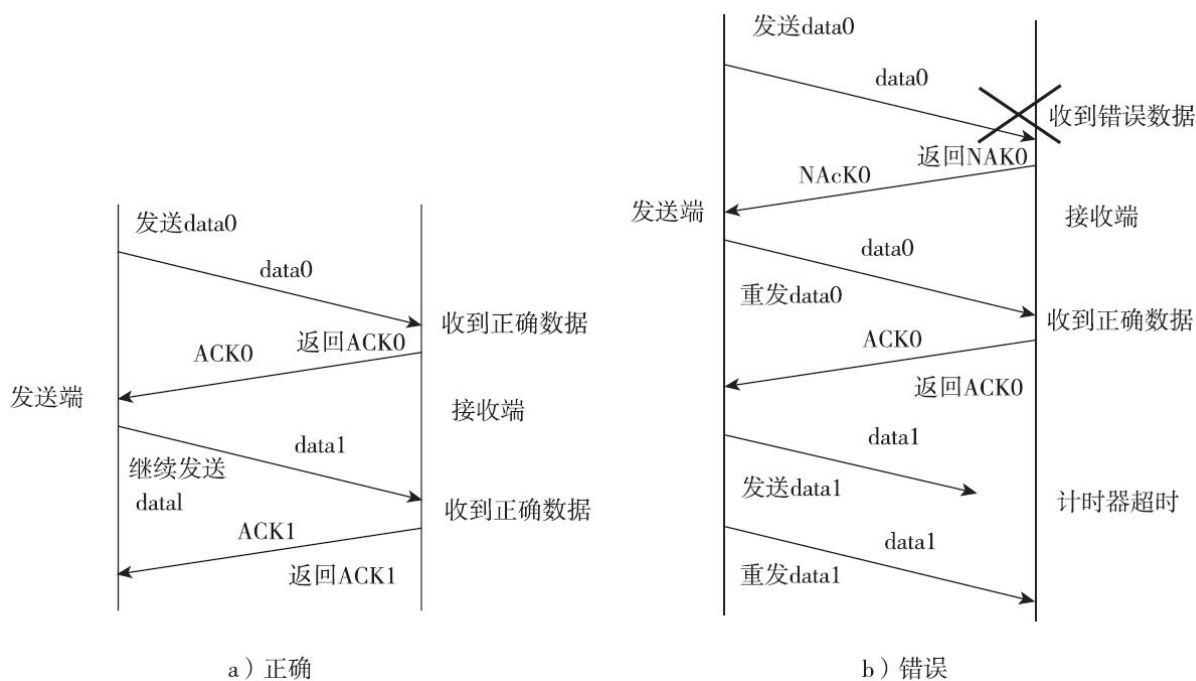


图 5-11 空闲重发请求方案原理示例

4) 后续数据帧的发送步骤与data0帧的发送一样。

现在来打个与这里所介绍的空闲重发请求差错控制方案类似的比方。一个牙牙学语的小孩，你要他从1数到10。由于他非常小，对这十个数字记得并不牢，所以他会一个个地去数，而且每数完一个数，他都会停下来，抬起头望着你，等待你的点头，或者说“对”，然后他才继续往下数。你对他点头或者说“对”，就相当于你向他发回了一个确认消息。

从以上过程可以看出，空闲重发请求方案除了要求在发送的数据帧中携带一定的检测错码外，还要求发送端和接收端都有一个用于存放待确认的发送帧和待向“网络层”提交的数据帧的缓冲存储空间，但这个存储空间比较小，因为它只需要临时存放一个数据帧。

由以上分析可以看出，这种ARQ方法设计简单，容易实现，但是这种方法也有一个不可克服的缺点，那就是每传送一个数据帧都要有一个等待时间（称为占空时间），信道的有效利用率低。占空时间与传送一个帧的全部时间的比例，称为占空比。数据帧越短，占空比就越大，相当于信道的利用率越低；数据帧越长，占空比就越小，信道的利用率就越高。但是数据帧越长，出错的概率也就越大，从而出现多次重发，因此传输效率也不会太高。正因如此才有下面将要介绍的连续重发请求差错控制方案。

5.3.5 连续重发请求方案

为了减小占空比，提高传输效率，人们又提出了连续重发请求（Continuous ARQ）的差错控制方案。

连续重发请求方案是指发送端可以连续发送一系列数据帧（也不总是不断地发送，具体可以连续发送多少个帧，要视双方的缓存空间大小，即窗口大小而定），即不用等前一帧被确认便可继续发送下一帧，效率大大提高。当然，在这个连续发送的过程中也可以接收来自接收端的响应消息（可以是确认帧，也可能是否认帧），发送端同样可以对传输出错的数据帧（如接收端返回了否认帧，或者响应计时器超时的帧）进行重发，具体处理方法后面会具体介绍。

由于连续重发请求方案减少了等待时间，整个通信的吞吐量就提高了，但在这种重发请求方案中，需要在发送端设置一个较大的缓冲存储空间（称为重发表），用以存放若干待确认的数据帧。当发送端得到某数据帧的确认帧后，便可从重发表中将该数据帧删除。

当出现传输差错时，连续重发请求方案有两种处理策略，即回退N帧（GO-BACK-N）策略和选择重发（Selective Repeat）策略。下面分别予以介绍。

1.回退N帧策略

回退N帧策略的基本原理是，如果发送端一共发送了n个数据帧（编号从0，一直到n-1），但收到接收端发来的ACK确认帧中少了某一个或几个帧的ACK确认帧（要么是数据帧出现了丢失，要么是这些帧对应的ACK帧或者NAK帧出现了丢失，最终造成ACK确认帧序号不连续），或者在接收某一帧时检测出有错，接收端发送一个NAK否认帧给发送端；或者在计时器超时后仍没有收到某个帧的ACK或者NAK帧，则发送端可以判断接收端最后一个正确接收的帧编号，然后从缓存空间的重发表中重发所收到的最后一个ACK帧序号以后的所有帧。

图5-12所示是一个回退N帧差错控制的示例。图中假定发送完8号帧后，0号和1号帧的ACK帧已收到，但2号帧的ACK帧在计时器超时后还未收到，则发送端只能退回，从2号帧开始重发以后所有已发的数据帧（2~8号共7个帧），尽管或许后面3~8号帧的ACK确认帧已收到。当然原来已发的这些帧接收端会自动删除的。

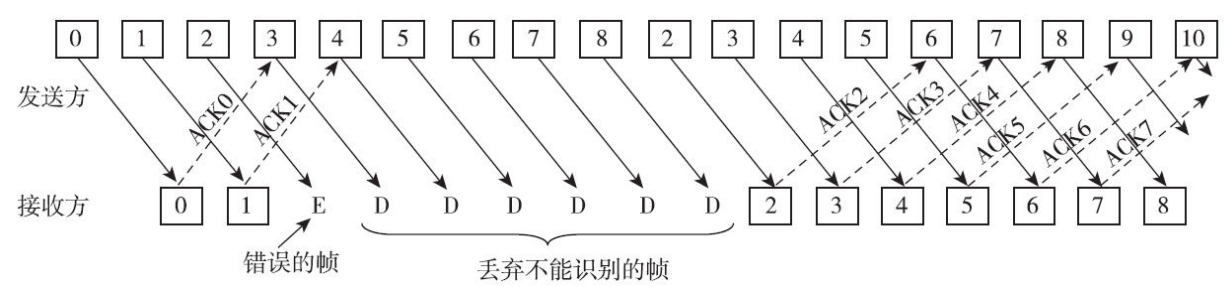


图 5-12 回退N帧策略差错控制原理示意图

下面我们将逐层深入分析“回退N帧”策略中的数据处理流程。

(1) 理想情形下的数据处理流程

首先看一下在理想状态的情形，也就是数据帧和确认帧都不发生差错或丢失的情形（也就是不存在NAK帧返回，不存在数据帧和ACK帧丢失，也不存在计时器超时）下的数据处理流程：

- 1) 发送端连续发送数据帧，而不等待任何数据帧的ACK帧的返回；
- 2) 发送端在重发表中保存所发送的每个数据帧的备份；
- 3) 接收端对每一个正确收到的数据帧返回一个ACK帧，ACK帧中包括对应帧的编号；
- 4) 接收端保存一个接收（次序）表，包含最后正确收到的数据帧的编号；
- 5) 当发送端收到相应数据帧的ACK帧后，发送端即从重发表中删除该数据帧。

（2）存在帧差错情形下的数据流程

接下来考虑帧（包括数据帧和响应帧）出现差错的情形下回退N帧策略的数据处理流程，这里的“差错”既包括数据帧在接收端检测出的差错，也包括数据帧或响应帧在传输过程中出现丢失的差错。此时，回退N帧策略中的数据处理流程如下：

1) 假设发送的第 $N+1$ 个帧发生了差错，接收端要么检测出第 $N+1$ 帧有错，要么发现没有接收到 $N+1$ 帧，反而接收到了第 $N+2$ 帧或第 $N+3$ 帧，或后边其他帧；

2) 出现这种情况时，接收端立即返回一个相应的未正确接收的否认帧 $NAK(N+1)$ ，预示接收端最后正确收到的是第 N 帧（ $N+1$ 帧的前一帧）；同时对后面每个失序的数据帧，接收端都产生相应的 NAK 帧，否则如果所发送的 $NAK(N+1)$ 帧正好丢失或出错，将产生死锁，即发送端不停地发送新的帧，同时等待对第 $N+1$ 帧的确认，而接收端不停地清除后继的帧，当然可以通过超时机制或者流量控制来避免死锁的发生（如滑动窗口法，具体将在本章后面介绍数据链路层的流量控制原理时介绍）；

3) 发送端在收到 $NAK(N+1)$ 帧，或者收到了 $NAK(N+2)$ 、 $NAK(N+3)$ ……帧时，从重发表中重发第 $N+1$ 帧或者对应的 NAK 帧中序号所对应的帧，同时接收端清除所有失序的帧（从第 $N+2$ 帧或者对应 NAK 帧中序号所对应的帧开始，直到重新正确接收到重发的第 $N+1$ 或者对应 NAK 帧中序号所对应的帧）；

4) 接收端重新收到第 $N+1$ 帧，或者对应的 NAK 帧中序号所对应的帧，接收端就继续正常操作。

以上流程看似比较复杂，其实原理很简单。打个比方来说吧，比如我们经常要自己小孩从1数到100，以检验他的数数能力。小孩正在数着，你突然发现某个数数得不正确（如数到42时本应是43，却数成了53），或者中间漏了一个或多个数（如数到42时，本应是43却数成跳过了，从45开始数了），你就会要求小孩从出现错误的那个数开始重数（如从43开始数），尽管在后面大多数是数得正确的。这就是回退N帧策略的差错控制原理。

2.选择重发策略

回退N帧策略因可以连续发送数据帧而提高了传输效率，但也有不利的方面，那就是在重发时必须把原来已正确传送过的数据帧再次发送，仅仅是因为这些数据帧之前的某个数据帧或确认帧发生了差错，这样又使传输效率降低。所以当通信链路的传输质量很差、误码率较大时，回退N帧策略就没什么优势了，因为这时可能经常要重传大量的数据帧。

为了弥补回退N帧策略的不足，另一种效率更高的差错控制策略——选择重发策略诞生了。在这个差错控制策略中，当接收端发现某帧出错后，其后继续送来的正确帧虽然不能立即递交给接收端的“网络层”，但接收端仍可接收下来，先存放在一个缓冲区中，同时通过向发送端发送NAK否认帧，要求发送端重新传送出错的那一帧。一旦收到

重新发来的正确帧后，就可以与原已存于缓冲区中的其余帧一起按正确的顺序递交给网络层。

选择重发策略规定，当发送端收到包含出错帧序号的NAK帧时，据此序号从重发表中选出相应帧的备份，直接插入到发送帧队列的前面给予重发，因为重发表的帧重发是按照FIFO（先进先出）的机制进行排列的，插在前面是为了可以最先重发，避免了对后继正确数据帧的多余重发，使得传输效率明显提高了。

如果仍用小孩数数来打比方的话，就是你先让小孩自己一直数下去，发现数错时你记下来（不要打断他正常的数数流程），当小孩数完后你再根据这些错误要求小孩重数对应的数就行了，而不必要求他从错误的地方重新数下去。

下面也分两种情况来讨论“选择重发”策略的数据处理方法。

（1）数据帧出现差错情形下的处理流程

以下是当数据帧有差错（包括接收端检测到所接收的数据帧有差错，或者有数据帧丢失两种可能）时，“选择重发”策略的数据处理流程：

- 1) 发送端连续发送多个数据帧，接收端对每个已正确接受的数据帧返回一个ACK帧；现假设第N+1个帧出现差错或丢失如果检测到第

N+1个帧有错误，则向发送端返回一个否认NAK（N+1）帧；如果一直到收到第N+2个数据帧时还没收到第N+1帧，则表明该帧已丢失，接收端不产生任何动作；但无论结果如何，已正确接收的数据帧，如第N个帧、第N+2个帧、第N+3个帧.....仍会向发送方返回确认ACK帧；

2) 当发送端收到来自接收端的否定NAK（N+1）或者收到第N+2帧的ACK帧时，会检测出其失序（因为按顺序，在收到ACK（N+2）帧之前应该是收到ACK（N+1）帧），得知第N+1帧没有被确认。将第N+2帧从重发表中清除，并在继续发送后继数据帧之前重发第N+1帧。

图中5-13所示为由于接收端检测到2号帧有错，向发送端发送了一个否认帧NAK2，要求发送端选择重发2号帧的示意图。从中可以看出，“选择重发”策略下只需发送有错的帧，而不会发送从有错帧开始后面所的帧，显然减少了信道资源浪费，提高了传输效率，但要求发送端和接收端都有足够大的缓冲区空间，以便存储多个帧的重发表和预提交数据帧。

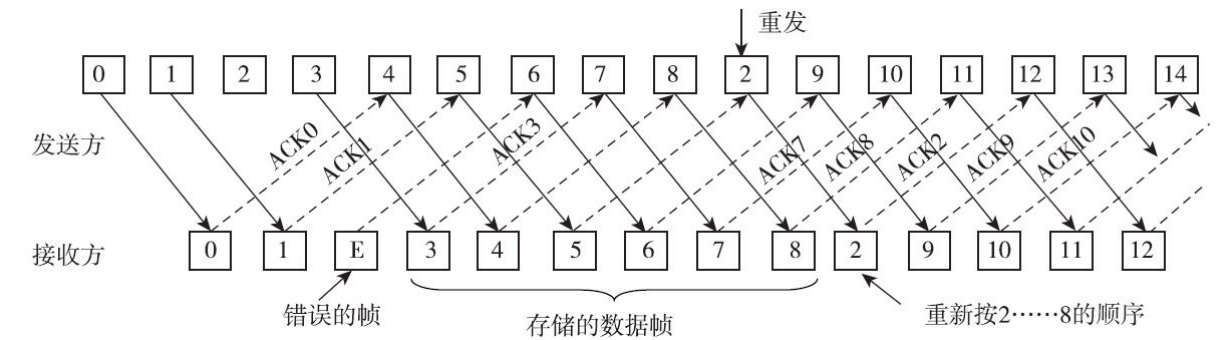


图 5-13 “选择重发”策略差错控制原理示意图

(2) 响应帧出现差错情形下的数据处理流程

在响应帧（包括确认ACK帧和否认NAK帧）出现差错时，也就是本应接收的是第N个帧的响应帧却接收到了第N+1个帧的响应帧情况下，在该情形下数据处理流程如下（现以ACK帧出错进行介绍，与NAK帧出错的处理流程一样）：

1) 当发送端已到了第N-1个帧的ACK帧，接下来应该收到的是第N个帧的ACK帧，而偏偏收到的是第N+1个帧的ACK帧；

2) 发送端在收到ACK (N+1) 帧后，检测出在重发表中第N个帧都还没收到ACK帧，因此认为第N个帧出现了差错（事实上并不是这样的），重发第N个帧；

3) 接收端在收到发送端重发的第N个帧，搜索接收表并确定第N帧已被正确接收，因此认定这个重发的第N帧是重复的，于是直接删除这个重发的第N帧，并返回一个ACK (N) 给发送端，以使发送端从重发表中删除第N帧。这样就达到了响应帧出现差错时的错误纠正。

5.3.6 海明纠错码

海明码（Hamming Code）是一个可以有多个校验位，具有检测并纠正一位错误代码功能的纠错码，所以它也仅用于信道特性比较好的环境中，如以太网中，因为如果信道特性不好的情况下，出现的错误通常不是一位。

海明码的检错、纠错基本思想是将有效信息按某种规律分成若干组，每组安排一个校验位进行奇偶性测试，然后产生多位检测信息，并从中得出具体的出错位置，最后通过对错误位取反（也是原来是1就变成0，原来是0就变成1）来将其纠正。

要采用海明码纠错，需要按以下步骤来进行：计算校验位数→确定校验码位置→确定校验码→实现校验和纠错。下面来具体介绍这几个步骤。

1. 计算校验位数

要使用海明码纠错，首先就要确定发送的数据所需要的校验码（也就是“海明码”）位数（也称校验码长度）。它是这样规定的：假设用N表示添加了校验码位后整个信息的二进制位数，用K表示其中有效信息位数，r表示添加的校验码位，它们之间的关系应满足：

$$N = K + r \leq 2^r - 1。$$

如 $K=5$ ，则要求 $2^r - r \geq 5 + 1 = 6$ ，根据计算可以得知 r 的最小值为4，也就是要校验5位信息码，则要插入4位校验码。如果有效信息位数是8，则要求 $2^r - r \geq 8 + 1 = 9$ ，根据计算可以得知 r 的最小值也为4。根据经验总结，得出信息码和校验码位数之间的关系如表5-1所示。

表 5-1 信息码位数与校验码位数之间的关系

信息码位数	1	2 ~ 4	5 ~ 11	12 ~ 26	27 ~ 57	58 ~ 120	121 ~ 247
校验码位数	2	3	4	5	6	7	8

2.确定校验码位置

上一步我们确定了对应信息中要插入的校验码位数，但这还不够，因为这些校验码不是直接附加在信息码的前面、后面或中间的，而是分开插入到不同的位置的。但不用担心，校验码的位置很容易确定的，那就是校验码必须是在 2^n 次方位置，如第1位，2位，4位，8位，16位，32位……（对应 2^0 ， 2^1 ， 2^2 ， 2^3 ， 2^4 ， 2^5 ，……，从最左边的位数起），这样一来就知道了信息码的分布位置，也就是非 2^n 次方的位置，如第3位，5位，6位，7位，9位，10位，11位，12位，13位，……（从最左边的位数起）。

举一个例子，假设现有一个8位信息码，即 b_1 、 b_2 、 b_3 、 b_4 、 b_5 、 b_6 、 b_7 、 b_8 ，由表5-1得知，它需要插入4位校验码，即 p_1 、 p_2 、 p_3 、 p_4 ，也就是整个经过编码后的数据码（称为码字）共有12位。根据以

上介绍的校验码位置分布规则可以得出，这12位编码后的数据就是 p1、p2、b1、p3、b2、b3、b4、p4、b5、b6、b7、b8。

假设原来的8位信息码为10011101，因还没有求出各位校验码值，现在这些校验码位都用“？”表示，最终的码字为：?? 1? 001? 1101。

3.确定校验码

经过前面的两步，我们已经确定了所需的校验码位数和这些校验码的插入位置，但这还不够，还得确定各个校验码值。这些校验码的值不是随意的，每个校验位的值代表了代码字中部分数据位的奇偶性（最终要根据是采用奇校验还是偶校验来确定），其所在位置决定了要校验的比特位序列。总的原则是：第i位校验码从当前位开始，每次连续校验i（这里是数值i，不是第i位，下同）位后再跳过i位，然后再连续校验i位，再跳过i位，以此类推。最后根据所采用的是奇校验还是偶校验即可得出第i位校验码的值。

（1）计算方法

校验码的具体计算方法如下：

p1（第1个校验位，也是整个码字的第1位）的校验规则是：从当前位数起，校验1位，然后跳过1位，再校验1位，再跳过1位，……。

这样就得出p1校验码位可以校验的码字位包括：第1位（也就是p1本身），第3位，第5位，第7位，第9位，第11位，第13位，第15位，……。然后根据所采用的是奇校验还是偶校验，最终可以确定该校验位的值。

p2（第2个校验位，也是整个码字的第2位）的校验规则是：从当前位数起，连续校验2位，然后跳过2位，再连续校验2位，再跳过2位，……。这样就得出p2校验码位可以校验的码字位包括：第2位（也就是p2本身），第3位，第6位，第7位，第10位，第11位，第14位，第15位，……。同样根据所采用的是奇校验还是偶校验，最终可以确定该校验位的值。

p3（第3个校验位，也是整个码字的第4位）的校验规则是：从当前位数起，连续校验4位，然后跳过4位，再连续校验4位，再跳过4位，……。这样就得出p4校验码位可以校验的码字位包括：第4位（也就是p4本身），第5位，第6位，第7位，第12位，第13位，第14位，第15位，第20位，第21位，第22位，第23位，……。同样根据所采用的是奇校验还是偶校验，最终可以确定该校验位的值。

p4（第4个校验位，也是整个码字的第8位）的校验规则是：从当前位数起，连续校验8位，然后跳过8位，再连续校验8位，再跳过8位，……。这样就得出p4校验码位可以校验的码字位包括：第8位（也就是p4本身），第9位，第10位，第11位，第12位，第13位，第14

位，第15位，第24位，第25位，第26位，第27位，第28位，第29位，第30位，第31位，.....。同样根据所采用的是奇校验，还是偶校验，最终可以确定该校验位的值。

.....

我们把以上这些校验码所校验的位分成对应的组，它们在接收端的校验结果（通过对各校验位进行逻辑“异或运算”得出）对应表示为G1、G2、G3、G4，.....（正常情况下均为0）。

（2）校验码计算示例

同样举上面的例子来说明，码字为 ?? 1 ? 001 ? 1101。

先求第1个“？”（也就是p1，第1位）的值，因为整个码字长度为12（包括信息码长和校验码长），所以可以得出本示例中p1校验码校验的位数是1、3、5、7、9、11共6位。这6位中除了第1位（也就是p1位）不能确定外，其余5位的值都是已知的，分别为1、0、1、1、0。现假设采用的是偶校验（也就是要求整个被校验的位中的“1”的个数为偶数），从已知的5位码值可知，已有3个“1”，所以此时p1位校验码的值必须为“1”，得出p1=1。

再求第2个“？”（也就是p2，第2位）的值，根据以上规则可以很快得出本示例中p2校验码校验的位数是2、3、6、7、10、11，也是一

共6位。这6位中除了第2位（也就是p2位）不能确定外，其余5位的值都是已知的，分别为1、0、1、1、0。现假设采用的是偶校验，从已知的5位码值可知，也已有3个“1”，所以此时p2位校验码的值必须为“1”，得出p2=1。

再求第3个“？”（也就是p3，第4位）的值，根据以上规则可以很快得出本示例中p3校验码校验的位数是4、5、6、7、12，一共5位。这5位中除了第4位（也就是p3位）不能确定外，其余4位的值都是已知的，分别为0、0、1、1。现假设采用的是偶校验，从已知的4位码值可知，也已有2个“1”，所以此时p2位校验码的值必须为“0”，得出p3=0。

最后求第4个“？”（也就是p4，第8位）的值，根据以上规则可以很快得出本示例中p4校验码校验的位数是8、9、10、11、12（本来是可以连续校验8位的，但本示例的码字后面的长度没有这么多位，所以只校验到第12位止），也是一共5位。这5位中除了第8位（也就是p4位）不能确定外，其余4位的值都是已知的，分别为1、1、0、1。现假设采用的是偶校验，从已知的4位码值可知，已有3个“1”，所以此时p2位校验码的值必须为“1”，得出p4=1。

最后就可以得出整个码字的各个二进制值码字为111000111101（带阴影的4位就是校验码）。

4.实现校验和纠错

虽然上一步已把各位校验码求出来了，但是如何实现检测出哪一位在传输过程中出了差错呢？（海明码也只能检测并纠正一位错误）它又是如何实现对错误的位进行纠正的呢？其实最关键的原因就是海明码是一个多重校验码，也就是码字中的信息码位可同时被多个校验码校验，然后通过这些码位对不同校验码的联动影响最终可以找出是哪一位出错了。

(1) 海明码的差错检测

现假设整个码字一共是18位，根据表5-1可以很快得出，其中有5位是校验码，再根据本节前面介绍的校验码校验规则可以很快得出各校验码所校验的码字位，如表5-2所示。

表 5-2 各校验码校验的码字位对照表

码字中的位	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
对应的位	p1	p2	b1	p3	b2	b3	b4	p4	b5	b6	b7	b8	b9	b10	b11	p5	b12	b13
p1 校验的位	√		√		√		√		√		√		√		√		√	
p2 校验的位		√	√			√	√			√	√			√	√			√
p3 校验的位				√	√	√	√					√	√	√	√			
p4 校验的位								√	√	√	√	√	√	√	√			
p5 校验的位																√	√	√

注：不带阴影的勾位是对应的校验码所在的位，而带阴影的勾位是该校验码要进行校验的数据位。

从表中可以得出以下两个规律：

□所有校验码所在的位是由对应的校验码进行校验的，如第1位（只由p1校验）、第2位（只由p2校验）、第4位（只由p3校验）、第8

位（只由p4校验）、第16位（只由p5校验），……也就是这些位如果发生了差错，影响的只是对应的校验码的校验结果，不会影响其他校验码的校验结果。这点很重要，如果最终发现只是一个校验组中的校验结果不符，则直接可以知道是对应校验组中的校验码在传输过程中出现了差错。

□所有信息码位均被至少两个校验码校验了，也就是至少校验了两次。查看对应的是哪两组校验结果不符，然后根据表5-2就可以很快确定是哪位信息码在传输过程中出了差错。

海明码校验的方式就是各校验码对它所校验的位组进行异或运算，即：

$$G1=p1\oplus b1\oplus b2\oplus b4\oplus b5\oplus\dots$$

$$G2=p2\oplus b1\oplus b3\oplus b4\oplus b6\oplus b7\oplus b10\oplus b11\oplus\dots$$

$$G3=p3\oplus b2\oplus b3\oplus b4\oplus b8\oplus b9\oplus b10\oplus b11\oplus\dots$$

$$G4=p4\oplus b5\oplus b6\oplus b7\oplus b8\oplus b9\oplus b10\oplus b11\oplus\dots$$

$$G5=p5\oplus b12\oplus b13\oplus b14\oplus b15\oplus b16\oplus b17\oplus b18\oplus b19\oplus b20\oplus b21\oplus b11\oplus b23\oplus b24\oplus b25\oplus b26\oplus\dots$$

正常情况下（也就是整个码字不发生差错的情况下），在采用偶校验时，各校验组通过异或运算后的校验结果均应该是为0，也就是前面所说的G1、G2、G3、G4，……均为0，因为此时1为偶数个，进行异或运算后就是0；而采用奇校验时，各组校验结果均应是为1。

现在举一个例子来说明，假设传输的海明码为 **111000111101**（一共12位，带阴影的4位就是校验码），从中可以知道它有4个校验组：G1、G2、G3、G4，然而到达接收端经过校验后发现只有G4=1（也就是只有这组校验结果不等于0），通过前面介绍的校验规律可以很快地发现是G4校验组中的p4校位码（也就是整个码字中的第8位）错了（因为只有一组校验结果出现差错时，则肯定只是对应的校验位出了差错），也就是最终的码字变成了：1110000011101。

再假设G3、G4两个校验值都不为0，也就是都等于1。通过表5-2所示比较G3、G4两个校验组（注意本示例中码字长度一共才12位，只需要比较前12位）中共同校验的码位可以很快发现是b8，也就是第12位出现了差错，也就是最终的码字变成了：111000011100。

经验之谈 这里一定要注意，最终有多少个校验组出现差错也不是随意的，一定要结合实际传输的码字长度来考虑。如上例一共12位，如果换成了16位的码字，且当b9位出现差错时，则G1、G3、G4一定会同时出现错误，因为b9这个位是三个校验组同时校验的，只要它一出错，肯定会同时影响这三个校验组的值。同理，如果是b11位出现

了差错，因为它同时受G1、G2、G3、G4四个校验组校验，所以这四个校验组结果都将出现错误。

(2) 海明码的差错纠正

检测出是哪位出现了差错还不够，因为海明码具有纠正一位错误的能力，所以还需要完成纠错过程。这个过程的原理比较简单，就是直接对错误的位进行取反或者加“1”操作，使它的值由原来的“1”变成“0”，或由原来的“0”变成“1”（因为二进制中每一位只能是这二者之一）。

以上就是海明码的差错检测和差错纠正原理了，虽然比单纯的奇偶校验码复杂些，但只要理清了思路，还是比较简单的。

5.4 流量控制

介绍完复杂的数据链路层“差错控制”功能后，接下来介绍数据链路层的“流量控制”功能。其实在上节介绍差错控制功能时就提到了流量控制功能，因为一些差错控制方案本身就具有一定的流量控制功能，如前面介绍的空闲重发请求方案中规定，发送端每发完一个数据帧后把这个帧保存在缓存空间中，然后就停下来等待接收端发来的确认消息，然后才能继续发送下一帧，这就可以控制路中的数据流量。在连续重发请求方案中，虽然发送端可以一次发送多个数据帧，但是也不是没有限制的，因为发送端需要把每次发送的数据帧保存在缓存空间中，接收端也要把向网络层提交的数据帧先保存在缓存空间中，所以最终一次能发送多少个帧，是由双方缓存空间大小决定的。这就是本节后面将要提到的“窗口大小”。

数据链路层的流量控制方案主要有两种：一种是适用于面向字符的异步通信协议（如RS——232）中的简单流量控制方案——XON/XOFF（继续/停止）方案；另一种是适用于大量数据通信环境中的“滑动窗口机制”。

5.4.1 XON/XOFF流量控制方案

XON/XOFF (transmitter on/transmitter off, 继续传输/停止传输) 是一种流量控制协议, 常用于数据传输速率大于等于1200bps, 而接收端数据处理速率远小于这个值 (也就是通信双方速率不同步) 的情形, 通过对发送端的数据传输速率进行控制, 以达到与接收数据数据处理速率匹配。

XON/XOFF (继续/停止) 是一种最简单的流量控制技术, 主要用于异步通信中, 接收端通过使用特殊字符来控制发送端数据的发送。其基本思想是: 当接收端认为不能继续接收数据时 (也就是接收端的缓存空间满了或者接近满时), 接收端会向发送端发送一个 **XOFF** 控制字符, 当发送端收到对应的 **XOFF** 控制字符时就停止数据的继续发送; 当接收端可以继续接收数据时, 接收端会再向发送端发送一个 **XON** 控制字符, 发送端收到这个控制字符后就知道可以恢复数据发送了, 继续发送数据, 一直这么循环下去。

其中 **XON** 采用 **ASCII** 字符集中的控制字符 **DC1** (十进制值为17, 十六进制值为11, 相当于按下“**Ctrl+Q**”组合键), **XOFF** 采用 **ASCII** 字符集中的控制字符 **DC3** (十进制值为19, 十六进制值为13, 相当于按下“**Ctrl+S**”组合键)。在一次数据传输过程中, **XOFF**、**XON** 的周期可重复多次, 但这些操作对用户来说是透明的, 也就是说用户不用管它, 设备会自动操作。

许多异步数据通信软件均支持XON/XOFF协议。这种方案也可用于计算机向打印机或其他终端设备（如Modem的串行通信）发送字符，在这种情况下，打印机或终端设备中的控制部件用以控制字符流量。如我们在通过Modem拨号连接网络时，采用的就是这种流量控制方法。当从PC机上的数据到达Modem时，如果Modem中的缓存空间满了，它就会向PC机发送一个代表“停止传输”的XOFF控制字符，而当Modem中的缓存空间没满时，Modem又会向PC机传送一个代表“继续传输”的XON控制字符。

再如，在局域网中一台PC机连接了一台打印速度比较慢的打印机，当PC机开始向打印机发送要打印的文件时，因为PC机的数据传输速率非常高，有许多文件打印机很难及时打印。此时打印机会向PC机发送一个XOFF字符来通知PC机，要求暂停文件的发送。PC机上的软件看到来自打印机的XOFF控制字符后，就会暂停文件的发送；而当打印机中排队等候打印的文件打完了，或者打印得差不多了时，打印机又会向PC机发送一个XON控制字符，通知PC机可以继续发送要打印的文件。

5.4.2 滑动窗口机制

在上面介绍的XON/XOFF流量控制方案中，为了实现发送端与接收端的速率匹配，需要往返发送一些特殊的控制字符，这样就会使得信道的利用率大打折扣，其主要是用于与一些低数据处理能力的接收端通信时采用。在实际的数据链路层流量控制中，更多的是采用本节将要介绍的“滑动窗口机制”来进行流量控制的（传输层也有这样的流量控制方案，具体将在第10章介绍）。

1.理解“滑动窗口”机制

“滑动窗口机制”中的“窗口”是指发送端和接收端的缓存空间大小；“滑动”的意思是指缓存空间中存放的未处理帧数是变化的，发送端在收到确认帧后会删除原来保存在缓存中的待重发帧，而接收端向网络层提交一个帧后也会删除原来保存在缓存中的帧。

至于缓存空间大小，采取不同的流量控制方案其会有不同的值，但要明白的是，缓存空间都是非常有限的，就像计算机CPU中的缓存一样。缓存越大，成本越高。如在5.3.4节介绍的空闲重发请求方案中，一次只能发送一个帧，发完一个帧后就等待来自接收端的确认帧，收到确认帧后就删除原来保存在缓存空间中的待重发帧，接收端不需要缓存空间，因为它接收到数据后即进行处理，对于有错误的帧

直接丢弃。所以在“空闲重发请求”方案中仅发送端需要保存一个帧的缓存空间，也就是它的“缓存空间大小”就是一个帧。

而在前面介绍的“连续重发请求”方案中，发送端一次可以连续发送多个帧，并且在其缓存空间中保存所有已发，但没有接收到来自接收端确认帧的待重发帧，而不用像“空闲重发请求”那样发一帧停下来等待接收端的确认帧；接收端也可以在缓存空间中保存来不及处理的帧，大大提高了数据传输的效率。

同样打个简单的比喻来说。就像一个水缸接了大小两个不同口径的水管，进水管的口径较大，出水管的口径较小。在这样的情况下，进水管肯定不能持续不断地向水缸中加水，否则就会因出水速率不匹配导致水从水缸中流出来了。所以通常是进水管供了一段时间的水后就要停下来，等水缸中的水用得差不多了再加水。这时水缸的容量就相当于上面所说的接收端缓存空间大小了。

这里涉及一个非常重要的问题，那就是到底一次最多连续发多少个帧比较合适呢？这里要考虑两方面的因素，一是要能实现有效的差错控制，二是要与接收端的数据处理能力相匹配，前者我们已介绍，本节仅从后者来考虑，也就是从流量控制角度来考虑。缓存空间大小又与帧编号有关，因为在数据传输时，每个帧都是有序列号的，这在本章前面介绍“差错控制”方案中就已说到。缓存空间越大，用于帧编号的位数就要越多，如1位可以表示2个帧（也就是窗口大小为2），2

位可以表示4个帧（也就是窗口大小为4），3位可以表示8个帧（也就是窗口大小为8），……。但用于指示帧序列号的位数越多，帧的无用开销就越大，所以在一般的数据链路层协议中只有2位用于帧编号。

2.滑动窗口实例

在此以1位帧序列号，也就是窗口大小为2的示例来介绍“滑动窗口机制”。下面从初始状态开始介绍整个流程（注意，这里仅考虑正常传输情况，不考虑出现差错的情况），如图5-14所示（各步对应图中的相应序号）：

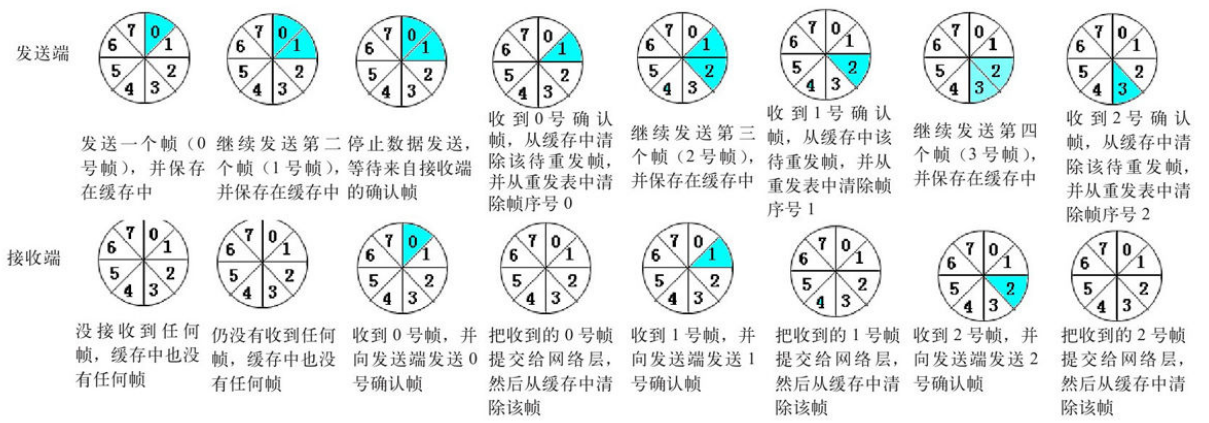


图 5-14 “滑动窗口机制”工作流程示例

- 1) 在初始状态下，发送端和接收端的缓存空间中均没有保存数据帧。
- 2) 发送端开始发送第一个帧——0号帧，并把它保存在缓存空间中，并建立一个待重发表，第一个帧号就是“0”。

3) 因为是1个比特位用于帧编号，窗口大小为2，所以发送端还可以继续发送第二个帧——1号帧。同时把这个帧保存在缓存空间中，并向待重发表中添加第二个帧的序号“1”。此时因为在发送端的缓存空间中已保存了两个帧（0号帧和1号帧），达到了窗口大小的值，不能继续发送后面的帧了，先停下来等待来自接收端的确认帧。在等待的过程中，接收端可能收到了0号帧，然后向发送端返回0号帧的确认消息。

4) 发送端在收到0号帧的确认帧后，立即从缓存空间中清除原来保存的0号待重发帧，并在待重发表中清除帧序号“0”；接收端把收到的0号帧提交给网络层，并清除缓存空间中的0号帧，此时缓存空间中又为空了。

5) 此时发送端继续发送第三个帧——2号帧，同时也把这个帧保存在缓存空间中，并向待重发表中添加第三个帧的序号“2”。因为此时发送端的缓存空间中又已保存了两个帧（1号帧和2号帧），又达到了窗口大小的值，不能继续发送后面的帧了，要再先停下来等待来自接收端的确认帧。在等待的过程中，接收端可能又收到了1号帧，然后向发送端返回1号帧的确认帧。

6) 发送端在收到1号帧的确认帧后，立即从缓存空间中清除原来保存的1号待重发帧，并在待重发表中清除帧序号“1”。接收端把收到

的1号帧提交给网络层，清除缓存空间中的1号帧，此时缓存空间中又为空了。

7) 此时发送端继续发送第四个帧——3号帧，同时也把这个帧保存在缓存空间中，并向待重发表中添加第四个帧的序号“3”。同样，因为此时发送端的缓存空间中又已保存了两个帧（1号帧和2号帧），又达到了窗口大小的值，不能继续发送后面的帧了，再要先停下来等待来自接收端的确认帧。在等待的过程中，接收端可能又收到了2号帧，然后向发送返回2号帧的确认帧。

8) 发送端在收到2号帧的确认帧后，立即从缓存空间中清除原来保存的2号待重发帧，并在待重发表中清除帧序号“2”。接收端把收到的2号帧提交给网络层，清除缓存空间中的2号帧，此时缓存空间中又为空了；

.....

后面的步骤按照以上流程类推。

说明 数据链路层中常见的协议可分为两大类：面向字符的链路层协议和面向比特的链路层协议（这些都属于同步传输模式协议）。面向字符的链路层协议主要有IBM BSC（Binary Synchronous Communication，二进制同步通信）协议、DEC DDCMP（Digital Data Communications Message Protocol，数字数据通信消息协议）、SLIP

（Serial Line Internet Protocol，串行线路网际协议）、PPP（Point-to-Point Protocol，点对点协议）等；面向比特的链路层协议主要有IBM的SDLC、ANSI通过修改SDLC而提出的ADCCP（Advanced Data Communication Control Procedure，高级数据通信控制规程）、ISO通过修改SDLC而提出HDLC（High-level Data Link Control，高级数据链路控制）、CCITT通过修改HDLC而提出LAP（Link Access Procedure，链路访问规程）等。下面将介绍几种典型的数据链路层协议。

5.5 面向字符的BSC协议

面向字符的同步方法也称“字符填充的首尾定界符法”。在该同步方法中，数据帧中的数据都被看作字符序列（所以称之为面向字符的同步传输），所有的控制信息也都是字符形式（当然数据的表示形式还是二进制的比特流），每个数据块的头部用一个或多个同步字符SYN来标记数据块的开始；尾部用字符ETX来标记数据块的结束。

面向字符的同步传输协议的典型代表就是IBM公司的BSC协议。BSC协议规定，链路上传送的数据必须是由规定字符集（可以是ASCII，或者EBCDIC（Extended Binary Coded Decimal Interchange Code，扩展二进制-十进制交换码））中的字符组成，控制信息也必须由同一个字符集中的若干指定的控制字符构成。

5.5.1 BSC控制字符和数据块结构

BSC协议与所有同步传输协议一样，也是一次可以传送由若干个字符组成的数据块（通常是一帧），而不是一次只传送一个字符。同时规定了十种特殊字符（称为通信控制字符）作为这个数据块的开始与结束标志，以及整个传输过程的各种控制信息标志（并不是每个数

据块中都有这十种全部的控制字符)。这十种通信控制字符说明如下:

□ACK (Acknowledge): 确认标志, 由接收端发出的, 作为对正确接收到报文的响应。

□DLE (Data Link Escape): 转义标志, 用于指示后面的字符是数据字符, 而不是特殊控制字符。这是用来进行透明传输的, 当在报文中也存在这十个控制字符时, 在这些字符前加上DLE字符后, 通知接收端把它们当作普通的报文处理, 而不是作为控制字符来识别。具体将在本节后面介绍。

□ENQ (Enquire): 询问标志, 用于请求远程站点给出响应。响应可能包括远程站点的身份或状态。

□EOT (End of Transmission): 发送完毕标志, 用于表示一个或多个文本的发送结束, 并拆除链路。

□ETB (End of transmission Block): 块终止或组终止标志, 用于标志每个数据块的结束位置。仅在一个报文要分成多个数据块传输时才有此标志。

□ETX (End of Text): 文本终止标志, 标志报文文本的结束。仅在一个报文不分成多个数据块传输时才有此标志。

□NAK (Negative Acknowledge)：否认标志，由接收端发出的，作为对未正确接收的报文响应。

□SOH (Start of Head)：报头开始标志，用于表示报文的标题信息或报头的开始。仅在报文的第一个数据块中才有此标志。

□STX (Start of Text)：文本开始标志，标志标题信息的结束和报文文本的开始。每个数据块均有此标志。

□SYN (Synchronous)：字符同步标志，用以实现通信双方的字符同步，或用于在无数据传输时保持同步。在每个数据块中均有此标志，而且通常是两个。

以上这十种通信控制字符所对应的ASCII码（ASCII中是用低7位表示一个字符的，最高位为校验码）或EBCDIC码值如表5-3所示。这些控制字符代码所对应的ASCII也可参见图4-7。这种通信控制字符中，在数据同步传输中起关键作用的就是SYN、SOH、STX、ETB、ETX、EOT这六种通信控制字符。

表 5-3 BSC 协议十种通信控制字符对应的代码

控制字符名称	对应的 ASCII 码	对应的 EBCDIC 码
ACK	0000110	00101110
DLE	0010000	00010000
ENQ	0000101	00101101
EOT	0000100	00110111
ETB	0010111	00100110
ETX	0000011	00000011
NAK	0010101	00111101
SOH	0000001	00000001
STX	0000010	00000010
SYN	0010110	00110010

通过前面的介绍可以知道，根据同步传输中报文在数据块中所处的位置不同，BSC协议的数据块有所不同，具体有如下四种格式。

□不带报头的单块报文或分块传输中的最后一块报文的格式为：

SYN	SYN	STX	报文	ETX	BCC
-----	-----	-----	----	-----	-----

□带报头的单块报文的格式为：

SYN	SYN	SOH	报头	STX	报文	ETX	BCC
-----	-----	-----	----	-----	----	-----	-----

□分块传输中的第一块报文的格式为：

SYN	SYN	SOH	报头	STX	报文	ETB	BCC
-----	-----	-----	----	-----	----	-----	-----

□分块传输中的中间报文的格式为：

SYN	SYN	STX	报文	ETB	BCC
-----	-----	-----	----	-----	-----

从以上四种数据报文格式中可以得出BSC报文传输的如下两个基本特点：①BSC协议中所有发送的数据均跟在至少两个SYN字符之后，以使接收端能实现字符同步；②所有数据块在块终止标志符（ETX或ETB）之后还有块校验字符BCC（Block Check Character），校验是单字节的CRC（循环校验码）或双字节的CRC，校验范围从STX开始到ETX或ETB为止。

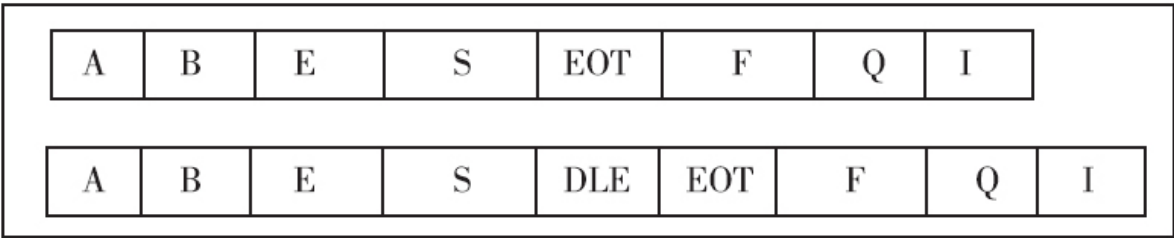
5.5.2 BSC协议数据透明传输原理

面向字符的同步传输协议不像异步传输协议那样需要在每个字符前后附加起始和停止位，因此传输效率提高了。但由于采用了一些特殊的传输控制字符，在增强了通信控制能力和校验功能的同时也带来了新的问题，那就是如何区别数据字符代码和特殊控制字符代码。因为在数据块中完全有可能出现与特殊控制字符代码相同的数据字符，这就会在接收端产生误解。比如正文有个与终止字符ETX的代码相同（在ASCII码中为0000011）的数据字符，如果不做任何处理，接收端就不会把它当作普通数据处理，而误认为是正文结束，进而产生差错。

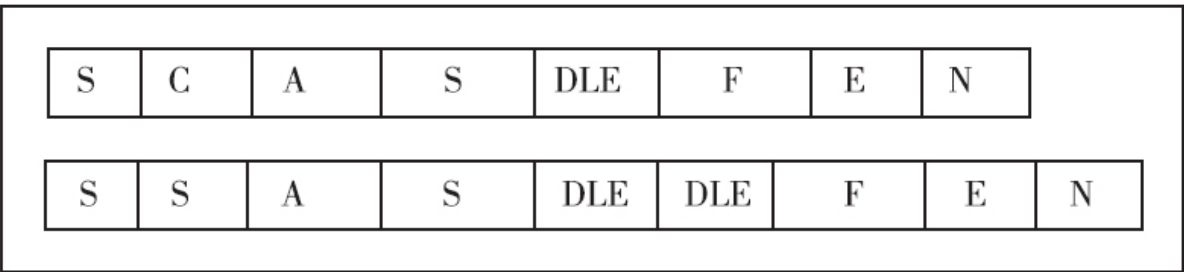
为此解决这一问题，面向字符的同步传输协议应具有将数据块中的特殊字符作为普通数据处理的能力，这种能力称为数据透明。解决方案是在同步传输协议中设置转义字符DLE（Data Link Escape，数据链路封装）。当需要在数据块中传输一个与某个特殊字符代码一样的数据时，就要先在它前面要加一个DLE代码（ASCII码中为0010000），这样接收端在收到一个DLE代码时，就知道了它下面一个字符是普通的数据字符，而不是通信控制字符。在接收端如果发现一个与某个特殊字符代码相同的字符是DLE代码，则直接删除前面的DLE代码，仅接收真实的数据部分。

如要在数据中传输EOT字符，为了避免把它误认为是通信控制字符，就是需先键入一个DLE字符，然后再键入EOT字符，如图5-15a所示。在接收端会直接删除EOT字符代码前面的DLE代码，仅接收数据部分的EOT代码。

同样，当要传输的数据中有一个代码与DLE的数据字符相同时，也要先在它前面加上另外一个DLE字符，相当于要传输一个普通的DLE数据字符，即需要键入两个“DLE”字符，如图5-15b所示。在接收端如果发现有连续两个DLE字符，则直接删除前面那个DLE字符，仅接收后面一个作为数据的DLE字符。



a) 当在数据中要传输 “EOT” 字符时



b) 当在数据中要传输 “DLE” 字符时

图 5-15 两个数据透明传输示例

从以上可以看出，这种面向字符的同步方法实现起来相当麻烦，因为BSC这类同步传输协议中的控制字符本身比较多，很容易就造成了与数据中传输数据代码的冲突。也正是因为这个缺点，后面又产生了下节将要介绍的面向比特的同步传输协议。

5.6 面向比特的SDLC和HDLC协议

面向比特的同步传输协议中，数据块是作为比特流来处理的（而不是作为字符流来处理），所以称之为面向比特的同步传输。在面向比特的同步传输中，每个数据块的头部和尾部都用一个特殊的比特序列（如01111110）来标记数据块的开始和结束，这就是面向比特的同步传输协议中的基本成帧原理，或者说是基本的帧同步原理。在局域网中所采用的传输方式都是面向位流的同步传输方式，由它们各自的介质访问控制协议来定义具体的数据格式（即帧格式）以及相应的介质访问控制方法。

面向比特同步传输的通信协议中，最具有代表性的是IBM的SDLC（Synchronous Data Link Control，同步数据链路控制）协议、国际标准化组织ISO的HDLC协议，美国国家标准协会ANSI的ADCCP。

SDLC协议是一种IBM数据链路层协议，适用于系统网络体系结构（SNA）；HDLC是一种在同步网上传输数据、面向位的数据链路层协议，是ISO对IBM的SDLC协议进行了改进和标准化的协议。但是SDLC属于单链路规程，而HDLC属于多链路规程。所有面向比特的数据链路控制（DLC）协议均采用统一的帧格式，且不论是数据还是单独的控制信息均以帧为单位传送。

SDLC支持识别两类网络站点：主站点（Primary Station）和从站点（Secondary Station）。主站点控制从站点，主站点轮询从站点是否有数据要发送。也就是说，从站点只有在主站点授权前提下才可以向主站点发送信息，如果一个从站点有数据要发送，当它被主站点识别后才可开始数据传输。主站点按照预先确定的顺序选择从站点，一旦选定的从站点已经导入数据，那么它即可进行数据传输。同时主站点可以建立和拆除链路，并在运行过程中控制这些链路。

5.6.1 HDLC链路结构和操作方式

HDLC是也采用主站点和从站点操作方式。在链路上起控制作用的站点称为主站点，其他的受主站控制的站点称为从站点。主站点负责对数据流进行组织，并且对链路上的差错实施恢复。由主站点发往从站点的帧称为命令帧，而由从站点返回主站点的帧称为响应帧。连有多个站点的链路通常使用查询技术，对其他站点进行查询的站点称为主站点，而在点到点链路中每个站点均可作为主站点。主站需要比从站点有更多的功能，所以当终端与主机相连时，主机一般总是主站点；在一个站点连接多条链路的情况下，该站点对于一些链路而言可能是主站点，而对另外一些链路而言又可能是从站点。

在HDLC中，对主站点、从站点和复合站点定义了图5-16所示的三种链路结构：不平衡链路结构、对称非平衡链路结构和平衡链路结

构。从图中可以看出，不同链路结构，允许的站点的类型以及各站点发送命令帧和响应帧的权限也不一样。

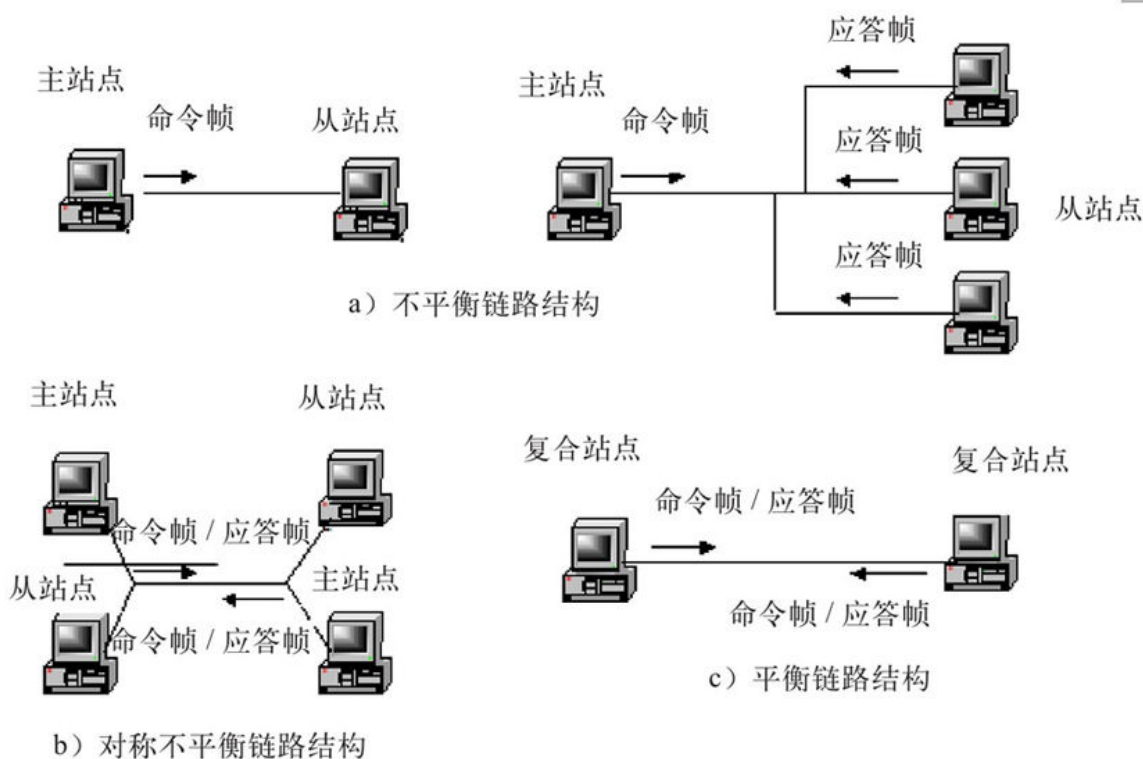


图 5-16 HDLC的三种链路结构

根据通信双方的链路结构和传输响应类型，HDLC提供了三种操作方式：正常响应方式、异步响应方式和异步平衡方式。

1.正常响应方式

正常响应方式（NRM）适用于不平衡链路结构，主要用于点对点 and 一点对多点的链路结构中，特别是一点对多点链路，一端为主站点，另一端为从站点，如图5-16a所示。在这种方式中，主站点控制着

整个链路的操作，负责链路的初始化、数据流控制和链路复位等，而从站点仅可在收到主站点的明确允许后，才能发出响应，所以它们的角色是不对称的，也就是不平衡的。

2.异步响应方式

异步响应方式（ARM）也适用于不平衡链路结构，它一般是对称不平衡链路结构，如图5-16b所示。ARM与NRM不同的是：在ARM中，从站点可以在没有得到主站点允许的情况下开始数据传输，所以它的传输效率比NRM高。

3.异步平衡方式

异步平衡方式（ABM）适用于平衡链路结构，即链路两端都是复合站点，也就是该站点既可作为主站点，又可作为从站点，如图5-16c所示。在这种链路结构中，两端的复合站点具有同等的能力，不管哪个复合站点均可在任意时间发送命令帧，并且不需要收到对方复合站点发出的命令帧就可以发送响应帧。ITU—TX.25建议的数据链路层就采用这种方式。

除了以上三种基本操作方式外，HDLC还有三种扩充方式，即扩充正常响应方式（SNRM）、扩充异步响应方式（SARM）、扩充异步平衡方式（SABM），它们分别与基本方式相对应。

5.6.2 SDLC/HDLC帧结构

下面以SDLC和HDLC协议为例介绍面向比特的同步传输协议的数据帧格式，如图5-17所示。各字段的说明如下：

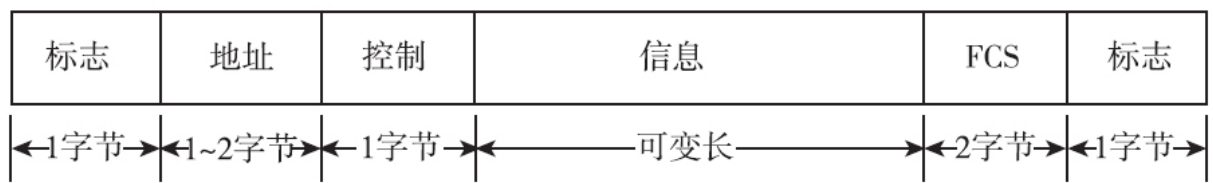


图 5-17 SDLC/HDLC帧格式

1.标志字段

SDLC/HDLC协议规定，所有信息传输必须以一个“标志”字段F（Flag）开始，且以同一个标志字段结束。也就是说每帧数据中有两个标志字段，值均为01111110（在EBCDIC码中是“=”字符），占1字节。标志字段用于界定不同帧，以获得帧边界，实现通信双方的帧同步，因为接收端可以通过搜索“01111110”来获知帧的开头和结束。通常，在不进行帧传送的时刻，为了维持信道处于激活状态，发送端会不断地发送标志字段，使接收端认为一个新的帧传送已经开始。

SDLC/HDLC中使用标志字段的方法可以是一个帧包括一个开始标志和一个结束标志（如图5-18a所示），也可以把一个帧的结束标志当成下一个帧的开始标志，也就是共享标志字段（如图5-18b所示），还

可以把一个帧的结束标志中的最后一个“0”当成下一帧开始标志的第一个“0”（如图5-18c所示）。

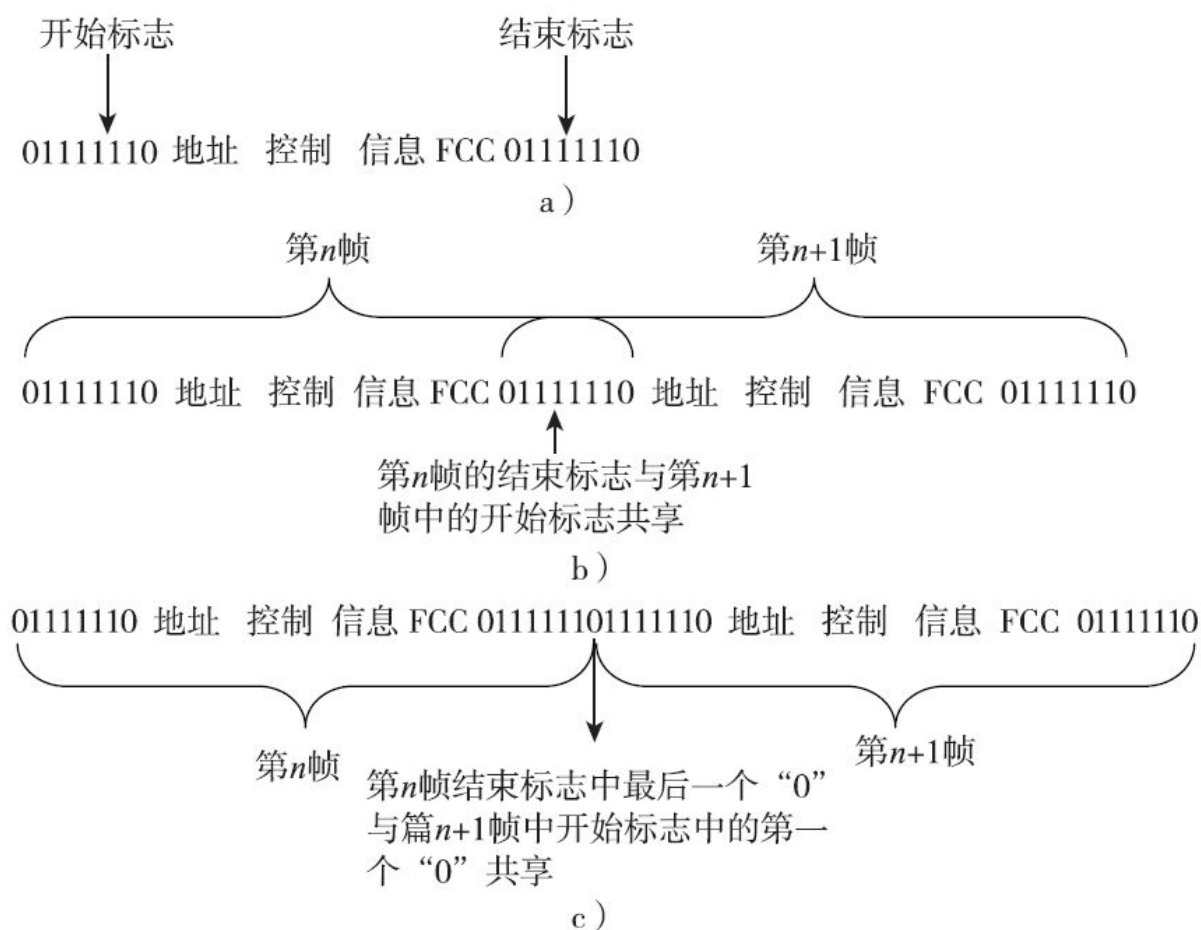


图 5-18 SDLC/HDLC协议标志字段的三种实现帧同步的方法

在一串数据比特中，有可能产生与标志字段的码型相同的比特组合——01111110。为了防止这种情况发生，保证对数据的透明传输，采取了比特填充技术。就是在信号码中出现连续5个“1”以后插入一个“0”，如图5-19所示。在接收端只需要再去掉5个“1”后面插入的“0”即可恢复原来的信号码序列。

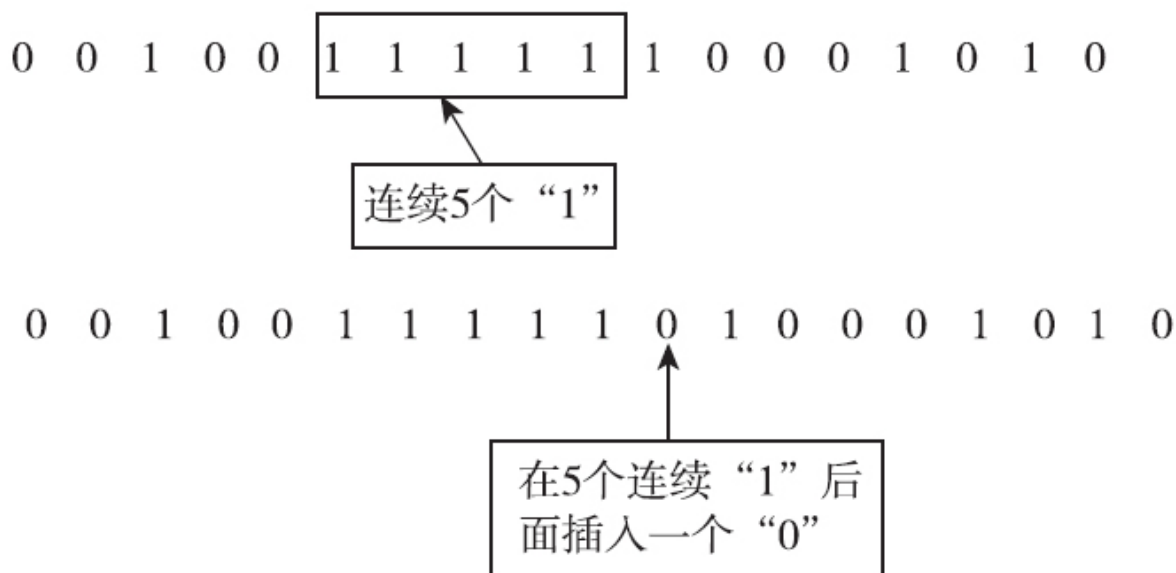


图 5-19 SDLC/HDLC的数据透明传输示例

2.地址字段

在“标志”字段之后是一个“地址”字段A（Address），表明帧是来自主站点还是来自从站点。主站点或者复合站点发送的命令帧中地址字段携带的是对方从站点的地址，而从站点发出的响应帧的地址字段携带的是本站点的地址。总体来说，在使用不平衡方式（包括NRM和ARM）传输数据时，地址字段总是写入从站的地址；在使用平衡方式（ABM）时，地址字段总是写入响应站点的地址。

“地址”字段可占8位或16位（也就是1，或者2个字节），通常是采用8位长度。00000000的地址为空地址，不能分配给任何站点，用于测试数据链路的状态；11111111地址为广播地址，使用这个地址就可以把一个数据帧发送到同一网段中其他所有站点。因此在采用8位地址格式

时，总的有效地址数是254个，这对一般的多点链路来说是足够了。但考虑在某些情况下，例如使用分组无线网，用户可能很多，可使用扩充地址字段，以字节为单位扩充。在扩充时，每个地址字段的第1位用作扩充指示，即当第1位为“0”时，表示后续一个字节为扩充地址字段；当第1位为“1”时，表示后续一个字节不是扩充地址字段，地址字段到此为止。

3.控制字段

“控制”字段C（Control）用来实现HDLC协议的各种控制信息，并标志是否是数据（因为它可以标志不同的帧类型），占1个字节长度。发送方主站点或复合站点利用控制字段来通知被寻址的从站点或复合站点执行约定的操作；而从站点用该字段对来自主站点或复合站点的命令帧进行响应，报告已完成的操作或状态的变化。

“控制”字段的结构如图5-20所示，共8位，根据其最前面两个比特的取值，可将HDLC帧划分为三大类，即信息帧I（Information）、监控帧S（Supervisory）和无编号帧U（Unnumbered）。如果控制字段中的第1位为0，则代表发送的是I帧；如果控制字段中的第1位和第2位为10，则代表发送的是S帧；如果控制字段中的第1位和第2位为11，则代表发送的是U帧。有关这三种帧类型及各自的“控制”字段具体结构，将在本节后面介绍。

	1	2	3	4	5	6	7	8
	b0	b1	b2	b3	b4	b5	b6	b7
I帧	0	N(S)			P/F	N(R)		
S帧	1	0	S		P/F	N(R)		
U帧	1	1	M		P/F	M		

图 5-20 HDLC“控制”字段结构

在“控制”字段结构中，最难理解的就是第5位的P/F（Poll/Final，查询/结束）位。在HDLC的各类帧中均带有这个P/F位，但在不同帧中的含义不一样：在由主站点发出的命令帧中取“P”位，起查询的作用，即当该位为1时，要求被查询的从站点做出响应；在从站点响应主站点的帧中取“F”位，起结束数据发送或确认结束的作用，即当该位为1时，表示从站点数据发送完毕，或者响应完毕。P/F位在不同的链路结构操作方式中所代表的含义并不一样。具体如下：

在NRM方式中，从站点不能主动向主站点发送信息，从站点只有在收到主站点发出P位为1（表示对各从站点依次进行查询，看看这些从站点是否有数据发送）的命令帧以后才能发送对应的响应帧：如果从站点有数据发送，则在最后一个I帧中将F位置1（表示数据发送结束），中间的I帧F位置0（表示数据还没发完）；如果从站点无数据发送，则要向主站点发送将F位置1的响应帧。

在ARM或ABM方式中，任何一个站点都可以在主动发送的S帧和I帧中将P位置1（同样是表示对各从站点或者对方复合站点依次进行查询，看看它们是否有数据发送）。对方站点在收到P=1的S帧或I帧后，同样会按照上面介绍的方法进行处理：如果对方站点有数据发送，则在最后一个S帧或I帧中将F位置1（表示数据发送结束），中间的S帧或I帧F位置0（表示数据还没发完）；如果对方站点无数据发送，则要向主站点发送将F位置1的响应帧。

4.信息字段

信息字段包含了用户的数据信息和来自上层的各种控制信息。在I帧和某些U帧中具有该字段，且可以是任意长度的比特序列。在实际应用中，其长度由收发站的缓冲器的大小和线路的差错情况决定，但必须是8位的整数倍。该字段可以是0长度，也就是可以无信息字段，如在监控帧（S帧）中就规定不能有信息字段。

5.帧校验序列字段

帧校验序列（FCS）字段可以使用16位的CRC，对两个标志字段之间的整个帧的内容进行校验。CCITT和ISO推荐使用的生成多项式为 $g(x)=x^{16}+x^{12}+x^5+1$ ；IBM SDLC使用的生成多项式为 $g(x)=x^{16}+x^{15}+x^2+1$ 。有关CRC的校验原理参见5.3.2节。

由于在SDLC/HDLC的帧中至少含有A（地址）、C（控制）和FCS（帧校验序列）这三个字段，所以整个帧长度最小为32位。

5.6.3 SDLC/HDLC帧类型及其标识方法

前面说了，在SDLC和HDLC等数据链路控制协议中，有I帧、S帧和U帧三种帧类型。下面具体介绍这些类型帧，以及在控制字段中如何标志它们。

1.信息帧（I帧）

I帧用于用户数据传输，包含信息字段。在I帧控制字段中第1位固定为0；第2位~第4位为N（S），用于标志要发送的帧的序号；第5位为P/F位；第6位~第8位为N（R），用于标志期待要接收的帧序号（它有两层含义：一是表示发送端已确认了前N-1个帧，另一个是发送端期待送第N帧的确认帧。）

N（S）和N（R）各占3位，即序号采用模8运算，使用0~7八个编号。在有些场合，如卫星通信中，模8已经不能满足要求了，这时可以把控制字段扩展为2字节，N（S）和N（R）都可用7位来表示，即增加到模128。

2.监控帧（S帧）

S帧用于监视和控制数据链路，完成I帧的接收确认、重发请求、暂停发送请求等功能。S帧没有信息字段（一共只有48位），可由主站

点或从站点发送，具体要视对应的链路结构操作方式。在S帧的控制字段中第1位和第2位分别固定为1、0；第3位和第4位为S，代表监控功能；第5位P/F和第6~8位与I帧一样。

因为用于表示监控功能的S共有2位，所以可以代表4种监控类型的帧，具体如下：

□00——接收就绪（RR），主站点可以使用RR型S帧来查询从站点，即希望从站点传输控制字段中第6~8位N（R）所指示的编号为N（R）的I帧，如果从站点存在这样的帧，便进行传输。从站点也可用RR型S帧来作为对主站点的响应帧，表示从站点希望从主站点那里接收的下一个I帧的编号是N（R）。

□01——拒绝（REJ），用于接收端要求发送端对从编号为N（R）开始的帧及其以后所有的帧进行重发，这也暗示N（R）以前的I帧已被正确接收。

□10——接收未就绪（RNR），用于接收端通知发送端编号小于N（R）的I帧已被收到，但目前正处于忙状态，尚未准备好接收编号为N（R）的I帧，这可用来对链路流量进行控制。

□11——选择拒绝（SREJ），用于接收端要求发送端发送编号为N（R）单个I帧，并暗示其他编号的I帧已全部确认。

上面四种S帧中，前三种用在回退N帧ARQ方案中，最后一种只用于选择重发ARQ方式中。有关这两种差错控制方案参见5.3.5节。

3.无编号帧（U帧）

U帧用于数据链路的控制，不带编号，可以在任何需要的时刻发出，而不影响带编号的信息帧的交换顺序。在U帧的“控制”字段中第1位和第2位都固定为1；第3位和第4位，以及第6~8位均为M，代表无编号功能；第5位P/F与I帧一样。

U帧可以分为命令帧（C）和响应帧（R），通过5个M位来表示不同功能，具体如表5-4所示。

表 5-4 无编号帧的类型及编码

帧名称	功能	帧类型		M 位	
		命令帧	响应帧	b ₂ b ₃	b ₅ b ₆ b ₇
SNRM	置正常响应模式	C		0 0	0 0 1
SARM/DM	置异步响应模式 / 断开方式	C	R	1 1	0 0 0
SABM	置异步平衡模式	C		1 1	1 0 0
SNRME	置扩充正常响应模式	C		1 1	0 1 1
SARME	置扩充异步响应模式	C		1 1	0 1 0
SABME	置扩充异步平衡模式	C		1 1	1 1 0
DISC/RD	断链 / 请求断链	C	R	0 0	0 1 0
SIM/RIM	置初始化方式 / 请求初始化方式	C		1 0	0 0 0
UP	无编号查询	C		0 0	1 0 0
UI	无编号信息	C		0 0	0 0 0
XID	交换识别	C	R	1 1	1 0 1
RESET	复位	C		1 1	0 0 1
FRMR	帧拒绝		R	1 0	0 0 1
UA	无编号确认		R	0 0	1 1 0

5.7 面向字符的PPP同步传输协议

前面介绍的BSC、SDLC、HDLC都属于局域网中的数据链路层协议，而本节要介绍的PPP（Point-to-Point Protocol，点对点协议）是一种应用非常广泛的广域网数据链路层协议。如我们在使用Modem进行拨号连接时就需要用到它，路由器设备间的Serial口之间的连接也要封装这个协议。本节要详细介绍这一协议的工作原理以及协议结构。

5.7.1 PPP简介

在点对点链路上，最早使用的数据链路层协议不是PPP，而是SLIP（Serial Line Internet Protocol，串行线路网际协议），如Windows 98系统中Modem拨号就是使用的SLIP，而不是PPP。但SLIP具有以下许多根本无法适应当时网络技术发展和应用需求的不足，具体如下：

□连接速率低：SLIP使用的线路速率一般介于1200bps和19.2kbps之间，远没有PPP的连接速率高（PPP最高可达128kbps）。

□不能自动分配IP地址：进行SLIP连接的通信双方必须先具备静态IP地址，不能在连接过程中动态分配IP地址。所以当时的SLIP通常应用于专线连接中，在拨号连接中也仅应用于固定IP地址方式的连接。

□无协议类型字段：在SLIP帧中没有协议类型字段，只支持默认的IP协议。SLIP帧很简单，只是在IP包的最前面和最后面各加一个END字符（0xc0），作为帧边界，标志一个帧的起始和结束。如果在包中有END字符，则包中的END字符用0xdb（ESC字符）和0xdc两个字符来替代；如果在包中有ESC字符，则ESC字符用0xdb（ESC字符）和0xdd两个字符来替代，以实现数据透明传输，如图5-21所示。

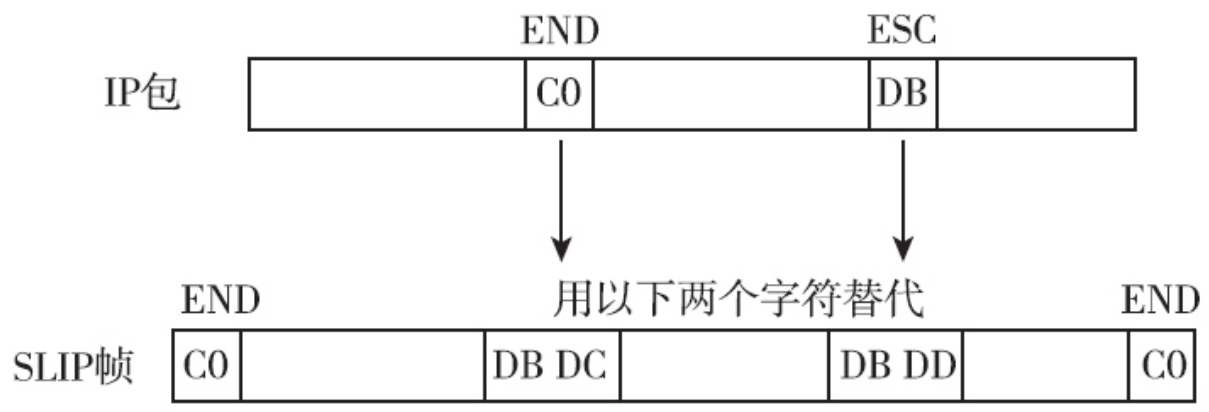


图 5-21 SLIP帧格式及透明传输示意图

□无帧校验序列（FCS）字段：从图5-21中可以看出，SLIP帧中没有FCS字段，因此在SLIP链路层上无法检测出传输差错，必须由上层实体或具有纠错能力的Modem来解决差错问题。

PPP协议是在SLIP的基础上发展起来的点对点数据链路层协议，对SLIP以上问题均加以了解决，但仍是面向字符的链路层协议。具体来讲，PPP协议具有以下几个方面的特性：

□PPP在连接速率上可以远高于SLIP，可以最高达到128kbps，如像v.90以上标准的Modem都可达到64kbps。

□提供了协议类型字段和帧校验序列（FCS）字段（如图5-22所示），使得PPP除了支持IP协议封装外，还可以封装其他三层协议包，如当时DECnet和Novell的Internet网包交换（IPX）。另外有了FCS字段，可以提供各种差错控制功能。

8	8	8	16	可变	16 ~ 32	8 位
标志	地址	控制	协议	信息	FCS	标志

图 5-22 PPP帧格式

□提供了一整套方案来解决链路建立、维护、拆除、上层协议协商、认证等问题。这些功能是通过以下几个子协议来完成的：

- 链路控制协议（Link Control Protocol，LCP）：用于建立、配置、测试和管理数据链路连接。
- 网络控制协议（Network Control Protocol，NCP）：协商该链路上所传输的数据包格式与类型，建立、配置不同的网络层协议。
- 口令认证协议（Password Authentication Protocol，PAP）和质询握手认证协议（Challenge-Handshake Authentication Protocol，CHAP）：为PPP连接提供用户认证功能，可以确保PPP连接的安全性。

家庭拨号上网就是通过**PPP**在用户端和运营商的接入服务器之间建立通信链路。在宽带接入技术日新月异的今天，**PPP**协议也衍生出新的应用。典型的应用是在**ADSL**（**Asymmetrical Digital Subscriber Loop**，非对称数据用户环线）接入方式当中，**PPP**协议与其他的协议共同派生出了符合宽带接入要求的新的协议，如**PPPoE**（**PPP over Ethernet**），**PPPoA**（**PPP over ATM**）。

利用以太网（**Ethernet**）资源，在以太网上运行**PPP**来进行用户认证接入的方式称为**PPPoE**。**PPPoE**既保护了用户方的以太网资源，又完成了**ADSL**的接入要求，是目前**ADSL**接入方式中应用最广泛的技术标准。同样，在下面将要介绍的**ATM**（异步传输模式，**Asynchronous Transfer Mode**）网络上运行**PPP**协议来管理用户认证的方式称为**PPPoA**。它与**PPPoE**的原理相同，作用相同；不同的是它在**ATM**网络上运行，而**PPPoE**在以太网网络上运行，所以要分别适应**ATM**标准和以太网标准。

PPP协议简单和功能完整的特点使它得到了广泛的应用，相信在未来的网络技术发展中，还可以发挥更大的作用。

5.7.2 PPP帧结构和透明传输原理

比较图5-22所示的PPP帧结构和图5-17所示的HDLC帧结构可以看出，它们是很相似的，其主要区别是PPP帧结构中多了一个协议字段（用来所封装的三层协议包类型），而且PPP是面向字符的协议，而HDLC是面向比特的协议，所以自然它们所采用的填充方式也不一样。下面先介绍PPP帧结构中各字段。

1.PPP帧结构

PPP帧结构参见图5-22，共分7个字段，其中标志字段在帧的最前面和最后面均有一个，其他字段各一个。下面是这些字段的具体含义说明。

□标志（Flag）：用来标志帧的起始或结束，占8位（1个字节），值固定为01111110（0x7E），与HDLC帧中的标志字段的值是一样的。

□地址（Address）：本来是用来标志对方节点地址的，但因PPP是点对点通信协议，是明确知道对方节点的，在实际通信中是无须知道对方的数据链路层地址（也就是MAC地址），从实际通信角度考虑，此地址字段实际上是没什么意义的，所以在PPP帧中此地址字段为固定的11111111（0xFF）标准广播地址，占8位（1个字节）。这与HDLC中的地址字段是不一样的。

□控制（Control）：因为PPP本身是一种可靠的点对点数据链路通信协议，无须像HDLC协议那样需要额外提供可靠的链路连接服务，所以PPP只有一种帧类型，那就是UI（无编号信息）帧。又因为它是无编号的帧，也就是无须接收端对收到的帧进行确认，所以，PPP帧中的控制字段其实也没有意义，值固定为00000011（0x03）。

□协议（Protocol）：PPP帧与HDLC帧的最大区别就是PPP帧中有协议字段，而HDLC帧中无该字段。之所以PPP帧中有协议字段，是因为它除了可以封装IP协议外，还可封装其他多种网络层协议包，如IPX、AppleTalk等。协议字段占16位（2个字节），指示在信息字段中封装的数据类型，如0x0021表示信息字段是IP数据包，0xC021表示信息字段是LCP（链路控制协议）数据，0x8021表示信息字段是NCP（网络控制协议）数据包，0xC023表示信息字段是PAP安全性认证数据包，0xC223表示信息字段是CHAP安全性认证数据包，0x0029表示信息字段为Apple Talk协议数据包，……

□信息（Information）：来自上层（“网络层”）的有效数据，可以是任意长度，默认为1500字节，如果不够该长度，还可以通过填充方法达到这个长度。

□帧校验序列（FCS）：使用16位的循环冗余校验计算信息字段中的校验和，以认证数据的正确性。

2.透明传输

从前面介绍的PPP帧结构中可以看出，在帧的首尾均有一个用于标志帧边界的标志字段，其值均固定为01111110（0x7E），这就同样要面对一个问题，那就是当在信息字段中出现和标志字段一样的比特0x7E时，接收端可能误把这些位当成帧边界。为了解决这个问题，也就是实现透明的数据传输，就必须采取一些措施。但因为PPP是面向字符协议，所以它不能采用HDLC所使用的零比特插入法，而是使用一种特殊的字符（也就是前面所说的转义字符）——0x7D进行填充。具体的做法是将信息字段中出现的每一个0x7E字节转变成2字节序列（0x7D，0x5E），即01111101 01011110；如果信息字段中出现一个0x7D的字节，则要将其转变成2字节序列（0x7D，0x5D）即01111101 01011101；如果信息字段中出现ASCII码的控制字符（如值为0x27的ESC字符），则在该字符前面要加入一个0x7D字节，如图5-23所示。这样做的目的是防止这些表面上的ASCII码控制字符被错误地解释为控制字符。

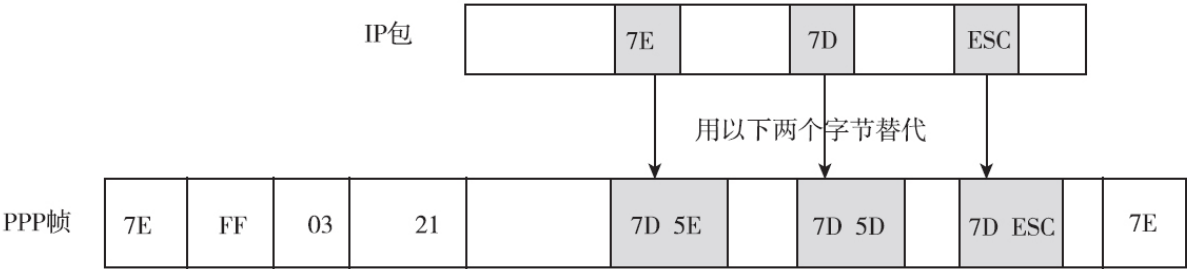


图 5-23 PPP帧格式及透明传输示意图

5.7.3 PPP链路建立、使用和拆除流程

在PPP通信中，因为不是像局域网中的链路那样始终连接的，所以在建立PPP通信前，通信双方必须协商建立链路连接，在链路建立后方可进行数据传输，数据传输完成后又可拆除原来建立的链路。整个过程分为五个阶段，即Dead（死亡）阶段、Establish（链路建立）阶段、Authenticate（身份认证）阶段、Network（网络控制协商）阶段和Terminate（结束）阶段。不同阶段进行不同协议的协商，只有前面的协议协商出结果后，才能转入下一个阶段协议的协商，如图5-24所示。

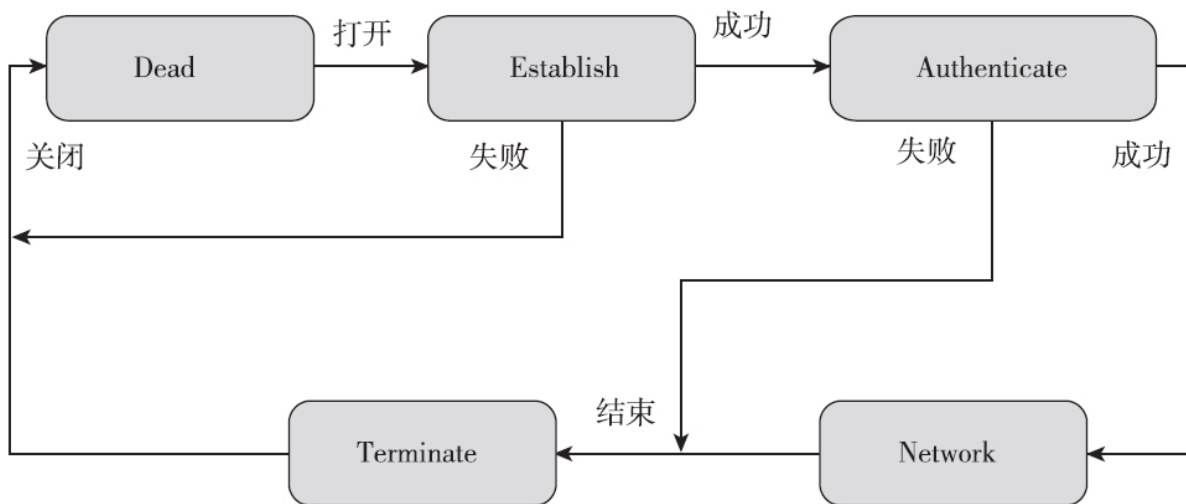


图 5-24 PPP链路建立、使用和拆除流程

PPP链路建立、使用和拆除的具体流程如下：

1) 当有用户向ISP或者对端节点发起PPP连接请求时，首先打开物理接口，然后PPP在建立链路之前先通过封装了LCP的PPP帧与接口进行协商，协商内容包括工作方式是SP（单PPP通信）还是MP（多PPP通信）、认证方式和最大传输单元等。

2) LCP协商完成后就进入Establish阶段，进行数据链路的建立。这时主要是启用PPP数据链路层协议，对接口进行封装。如果启用成功，则进入下一身份认证（Authenticate）阶段，并保持LCP为激活状态，否则返回关闭接口，LCP的状态为关闭。

3) 如果数据链路建立成功，则进入到Authenticate阶段，对请求连接的用户进行身份认证。具体要根据通信双方所配置的身份认证方式来确定是采用CHAP还是PAP身份认证。

4) 如果认证成功就进入Network阶段，使用封装了NCP的PPP帧与对应的网络层协议进行协商，并为用户分配一个临时的网络层地址（如IP地址）；如果身份认证失败，则直接进入Terminate（结束）阶段，拆除链路，返回到Dead阶段，LCP状态转为Down。

5) PPP链路将一直保持通信，直至有明确的LCP或NCP帧关闭这条链路，或发生了某些外部事件（如用户的干预），进入到Terminate阶段，然后关闭NCP协议，释放原来为用户分配的临时网络层地址，最后返回到Dead阶段，关闭LCP。

5.7.4 PPP的PAP/CHAP身份认证

在PPP通信中，可以采用PAP（密码认证协议）或者CHAP（质询握手认证协议）身份认证方式对连接用户进行身份认证，以防非法用户的PPP连接。如果双方达成一致，也可以不采用任何身份认证方式（如一般情况下的路由器间Serial口之间的PPP连接）。

1.PAP身份认证

PAP身份认证过程非常简单，是一个二次握手机制，整个认证过程仅需两个步骤：被认证方发送认证请求→认证方给出认证结果。

PAP身份认证可以在一方进行，即由一方认证另一方身份，也可以进行双向身份认证，也就是既要PAP服务器对PAP客户端的合法性进行认证，PAP客户端也需要对PAP服务器进行认证，以确保用于认证的PAP服务器是合法的。如果是双向认证，则要求认证的双方都要通过对方的认证程序，否则无法在双方之间建立通信链路。

下面以单向认证为例介绍PAP认证过程，如图5-25所示。但要注意的是，PAP认证是由被认证方（也就是PAP客户端）首先发起的。

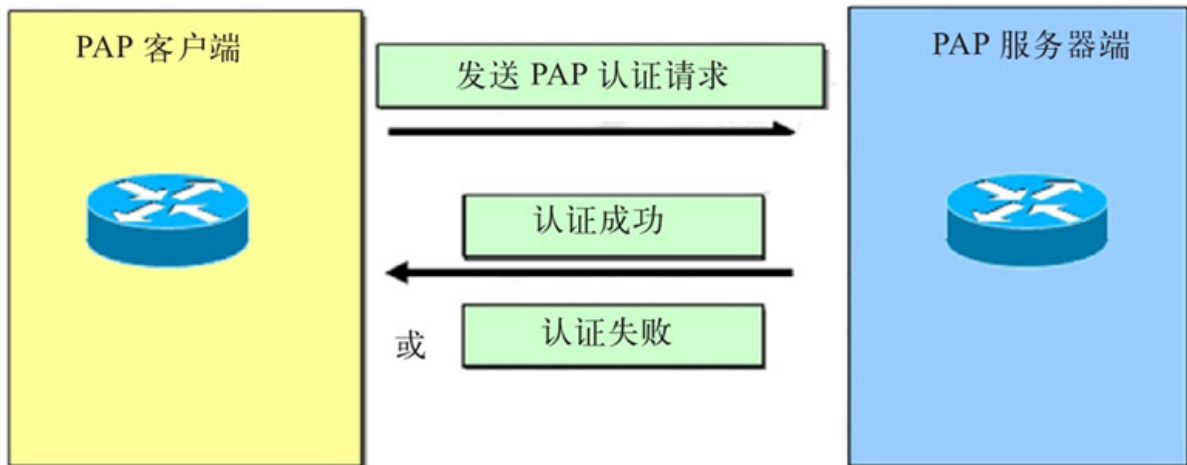


图 5-25 PAP身份认证的两次握手

1) 发起PPP连接的客户端（被认证方）首先向担当身份认证的PAP服务器端（如是向ISP拨号，则PAP服务器在ISP端，如果是路由器的串口对连，则PAP服务器必须要在对端设备上配置）发送一个认证请求帧，其中就包括用于身份认证的用户名和密码。

2) PAP服务器端（认证方）在收到客户端发来的认证请求帧后，先查看服务器本地配置的用户账户数据库，看是否有客户端提供的用户名/密码对信息（这个用户账户数据库必须先要在PAP服务器端配置好）。如果有，则表明客户端具有合法的用户账户信息，向PAP客户端返回一个认证确认（ACK）帧，表示认证成功，该用户可以与PAP服务器端建立PPP连接；否则返回一个认证否认（NAK）帧，表示认证失败，当然客户端也就不能与PAP服务器端建立PPP连接了。但这里要注意的是，如果第一次认证失败，并不会马上直接将链路关闭，而是会在PAP客户端提示可以尝试以新的用户账户信息进行再次认证，只有当

认证不通过次数达到一定值（默认为4）时才会关闭链路，以防止因误传、网络干扰等造成不必要的LCP重新协商过程。

PAP身份认证的特点是，用于身份认证的用户名及密码在网络上是以明文（也就是不加密）方式进行传输的，如在传输过程中被截获，便有可能对网络安全造成极大的威胁，所以**PAP**并不是一种安全有效的认证方法。

以上介绍的是**PAP**单向认证过程，仅两步（一问一答的形式），很简单。**PAP**双向认证过程与单向认证过程类似，只不过此时**PPP**链路的两端同时具有客户端和服务端双重角色，任何一端都可向对方发送认证请求，同时对对方发来的认证请求进行认证。

2.CHAP身份认证

CHAP认证过程相对前面介绍的**PAP**认证来说更为复杂，它采用的是三次握手机制（而不是**PAP**中的两次握手机制），整个认证过程要经过三个主要步骤：认证方要求被认证方提供认证信息→被认证方提供认证信息→认证方给出认证结果。

另外，**CHAP**身份认证方式相对**PAP**认证方式来说更加安全，因为在认证过程中，用于认证的用户名和密码不是直接以明文方式在网络上传输的，而是经过**MD5**之类的摘要加密协议随机产生的密钥；而且这个密钥是有时效的，原密钥失效后会随机产生新的密钥，所以即使

在通信过程中密钥被非法用户破解了，也不会适用于后面的通信截取。

与同PAP认证一样，CHAP认证也可以是单向或者双向的。如果是双向认证，则要求通信双方均要通过对对方请求的认证，否则无法在双方建立PPP链路。在此，我们仍以单向认证为例介绍CHAP认证流程，具体如图5-26所示。注意，CHAP身份认证首先是由CHAP服务器端发起的。

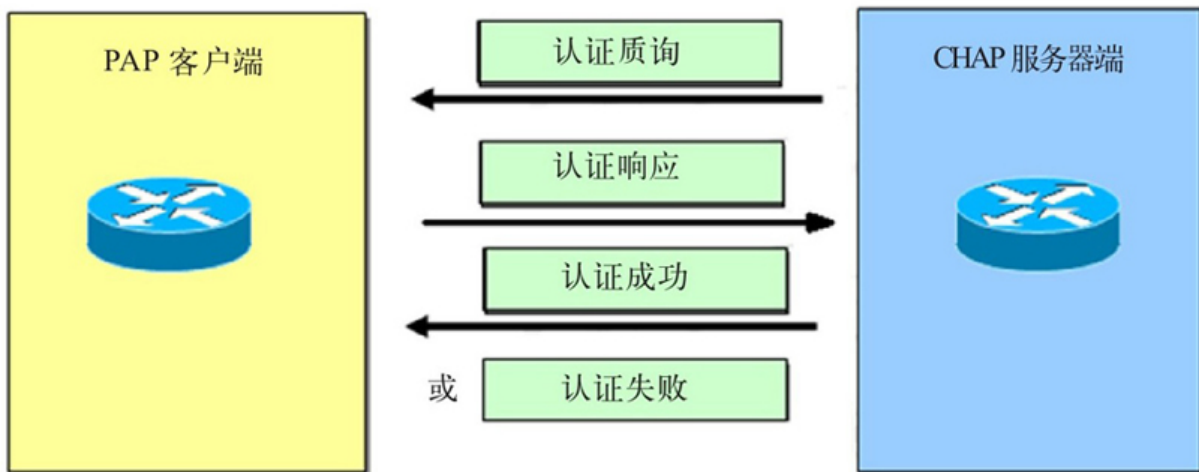


图 5-26 CHAP身份认证的三次握手

1) 当PPP链路建立起来后，采用CPAP身份认证方式时，首先是由CHAP服务器（认证方）不断地以产生一个随机序列号的质询（challenge，又称挑战）字符串帧发送给CHAP客户端（被认证方），询问客户端（被认证方）是否要进行身份认证。直到该客户端为这个质询做出了响应，也就是进入了下面的第2步。

2) 客户端在收到服务器端发来的质询消息后，把自己要用于身份认证的用户名和密码（这需要事先在客户端设备中配置好）通过MD5摘要加密协议生成一个随机序列的响应帧发送给服务器端。

3) 服务器端在收到来自客户端的响应后，同样利用MD5加密协议解密出其中的认证用户名和密码，然后在服务器本地用户账户数据库（也是需要事先在服务器端配置好的）中查找，看是否有相同的账户信息。如果找到一样的账户信息，表明请求认证的客户端是合法的，通过认证，允许客户端发起的PPP连接，否则拒绝认证，表示认证失败。与PAP一样，第一次认证失败后，也不会马上关闭链路，而是再次向客户端提示输入新的用户名和密码进行再次认证，直到规定的最高尝试次数。

CHAP认证方式的最关键一点就是采用了MD5摘要加密协议，用于认证的用户名和密码是直接在摘要消息中经过加密的，所以不会在网络中以明文方式传输，因此它的安全性要比PAP高。

5.8 数据链路层主要网络设备

介绍完数据链路层主要功能、服务后，下面我们再来简单介绍工作在数据链路层上的主要网络设备。在常见的网络设备中，如网卡、网桥和二层交换机。

5.8.1 计算机网卡

说到计算机网卡（也叫网络适配器），大家可能认为这没什么好说的，大家都见过、用过。网卡是安装在计算机上，用来连接计算机网络的，是计算机网络中最基础的网络设备。目前在计算机局域网中，网卡类型总的来说主要可分为有线以太网卡、WLAN无线网卡两大类。下面分别予以介绍。

1. 有线以太网卡

在企业局域网中，我们通常所说的有线网卡通常就是指以太网卡，所以在此也仅限于对以太网卡的介绍，而不再涉及目前在企业局域网中很少见到的其他网络类型的网卡，如令牌环网卡、令牌总线网卡和FDDI网卡等。另外，网卡除了要区分网络类型外，还可根据所应用的环境分为普通工作站网卡和服务器网卡两类。

有线网卡还可根据以下几个方面可以进行进一步划分：一是网卡的主机接口，也就是网卡与计算机的接口；二是网卡主机接口总线的位数；三是网卡的网络接口类型，也对应了网卡所支持的传输介质类型；四是网卡所支持的以太网标准。

在主机接口方面，有线网卡主要有以下几种类型：一是最常见的PCI总线接口，二是微型PCI（PCMCIA）接口，三是在服务器网卡用得比较多的PCI-X和PCI-E接口。这几种主机接口类型网卡如图5-27所示。

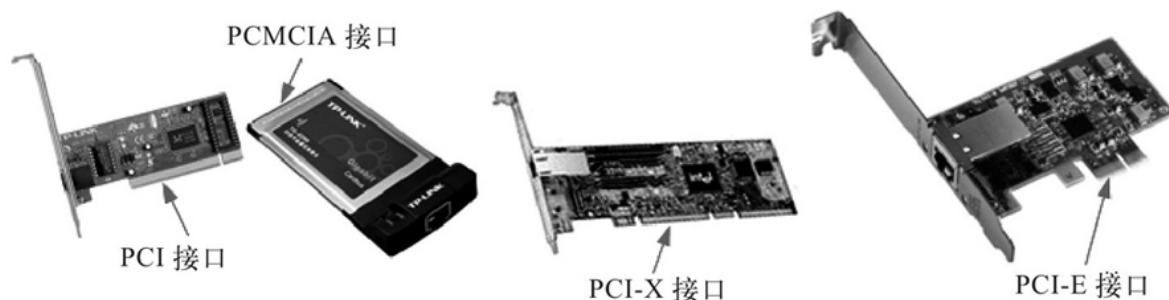


图 5-27 PCI、PCMCIA、PCI-X和PCI-E接口网卡

在有线网卡的网络接口方面，主要对应的是不同的传输介质。因为使用同轴电缆的以太网目前在企业局域网中比较少见了，所以在此不再介绍粗/细同轴电缆接口的以太网卡，仅介绍双绞线接口以太网卡和光纤接口以太网卡。

双绞线接口以太网卡使用最普遍，对应双绞网线连接器的接口为RJ-45类型。RJ-45连接器（俗称水晶头）和网卡上对应的RJ-45网络接

口如图5-28所示。

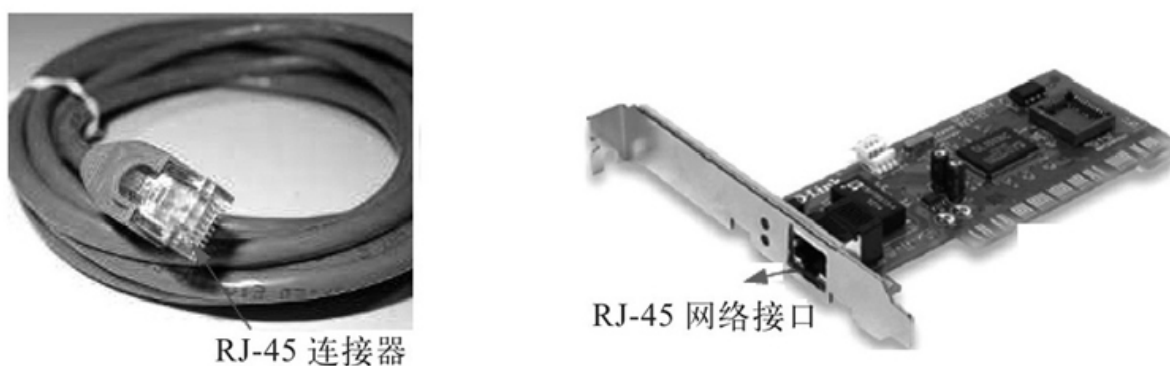


图 5-28 RJ-45连接器和网络接口

使用光纤作为传输介质的网卡所对应的光纤连接器接口有好几种，最常见的有ST、SC、FC、LC四种。这在本书第4章中有详细介绍，在此不再赘述。ST、SC、FC和LC这四种主要的光纤连接器接口如图5-29所示。图5-30所示是目前应用最广的两种光纤网络接口SC和LC的网卡示例。

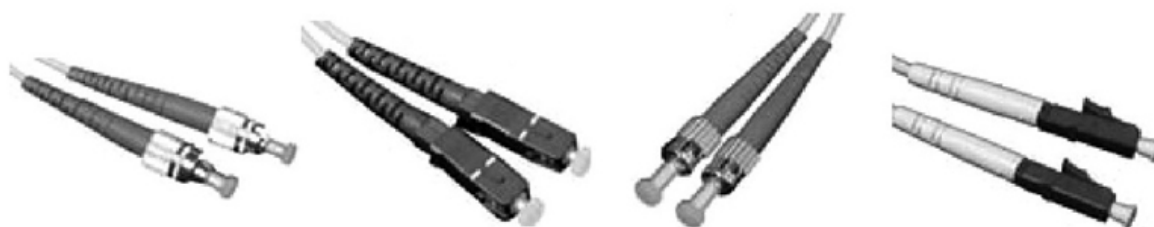


图 5-29 ST、SC、FC和LC光纤连接器接口

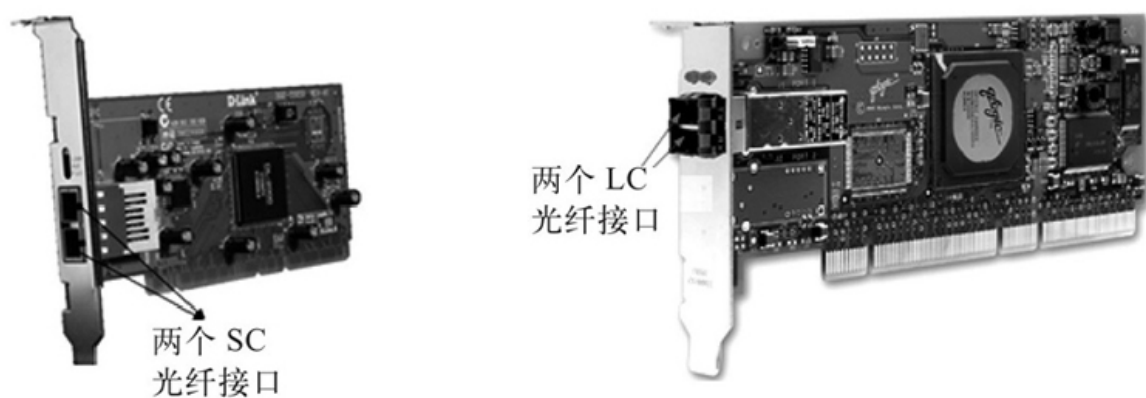


图 5-30 SC和LC网络接口网卡示例

有线网卡除了可以按主机接口和网络接口划分外，还可以根据网卡所支持的网络标准来划分。目前，在有线以太网工作站中我们通常采用支持10/100Mbps自适应的双绞线快速以太网卡，性能要求高一些的可能会用到支持自适应的10/100/1000Mbps双绞线千兆以太网卡。而服务器上的网卡目前基本上都是自适应的10/100/1000Mbps双绞线千兆以太网卡，或者纯1000Mbps的光纤千兆以太网卡。

另外，支持的这些不同网络标准的PCI接口以太网卡，所对应的主机接口工作模式也会有所不同。工作站上普遍使用的PCI接口快速以太网卡基本上都是32位的，而千兆以太网卡基本上都是64位的。要注意的是，纯64位的千兆以太网卡与向后兼容32位的64位以太网卡的PCI总线接口金手指结构（金手指长度和缺口数不一样）是不一样的。如图5-31a所示是32位的PCI接口快速以太网卡，图5-31b所示为纯64位的PCI

接口千兆以太网卡，图5-31c为同时支持32位和64位的PCI接口千兆以太网卡。

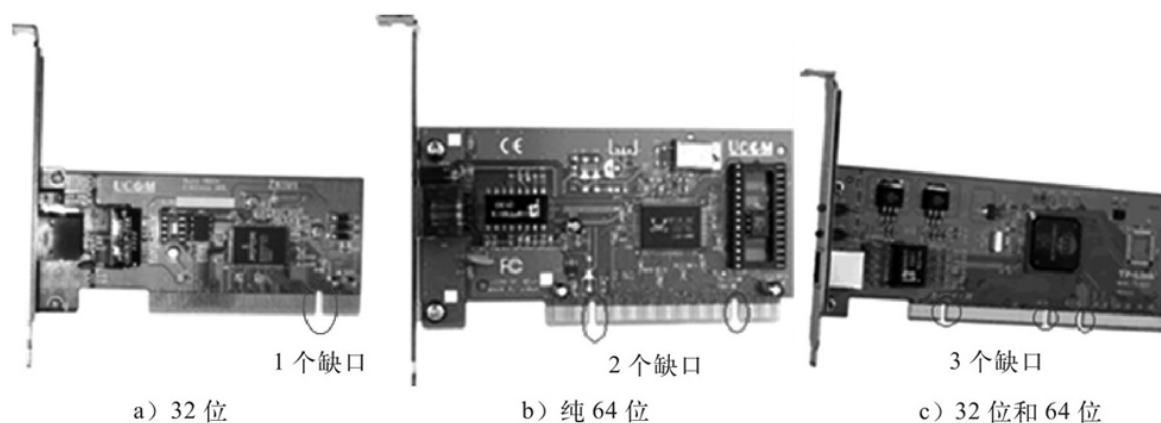


图 5-31 三种不同工作模式的以太网卡

2.WLAN网卡

在WLAN网卡方面，相对有线网卡来说要简单一些，那是因为就目前来说WLAN技术还不如有线网络那么先进，所以在主机接口方面没有什么特殊要求，一般的32位PCI或者USB 1.1、2.0或者3.0版本接口即可满足。

在WLAN网卡选择方面主要考虑网卡的主机接口和所使用的WLAN技术标准。在台式机工作站中通常选用PCI或者USB接口的WLAN网卡（主要是PCI接口），如图5-32所示。



图 5-32 PCI和USB接口的WLAN网卡

对于笔记本电脑用户则可以选择PCIMCIA和USB两种接口类型的无线局域网网卡。而PCMCIA接口又分16位的PCMCIA和32位的CardBus两种接口类型，如图5-33所示。在无线局域网标准上，目前至少建议应选择具有54Mbps速率的IEEE 802.11g标准无线网卡产品，普遍采用的是支持600Mbps速率的IEEE 802.11n标准的无线网卡。有关WLAN标准的内容将在下章介绍。



图 5-33 PCMCIA和CardBus接口WLAN网卡

5.8.2 网桥及其工作原理

网桥（**Bridge**）是早期的两端口二层网络设备，用来连接不同网段的计算机网络设备（如图5-34所示），同时它又可隔离冲突域，因为它的两个端口不是共享一条背板总线（分别有一条独立的交换信道），比当时的集线器（**Hub**）性能更好（集线器上各端口都是共享同一条背板总线的）。后来，网桥被具有更多端口、同时也可隔离冲突域的交换机（**Switch**）所取代。

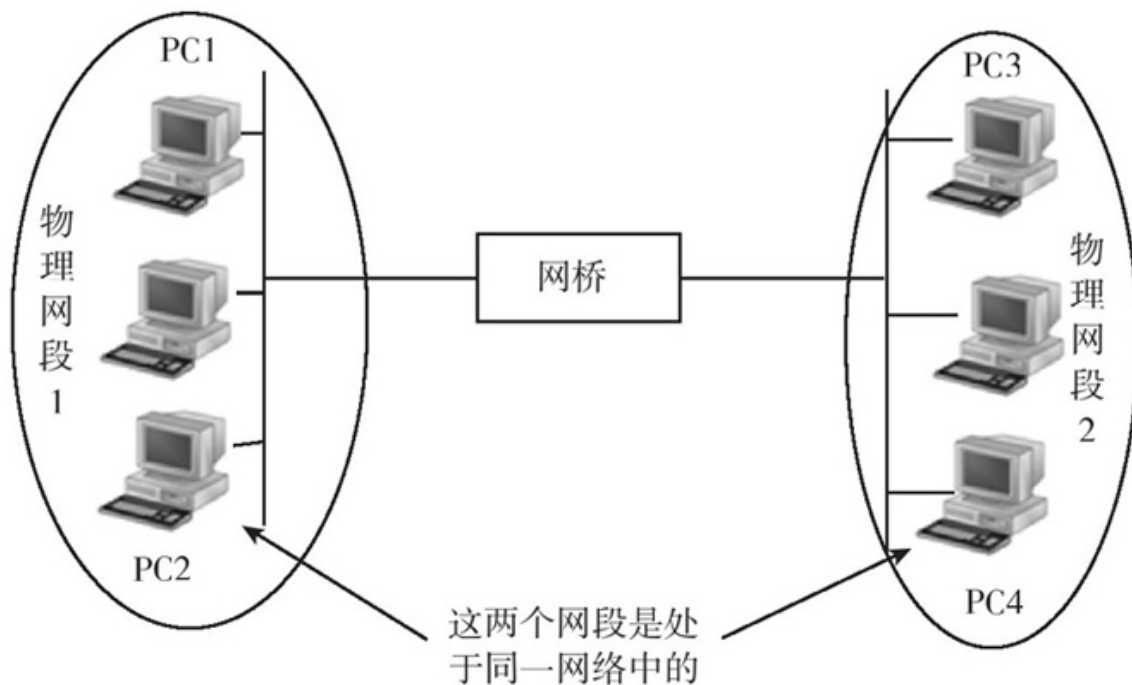


图 5-34 网桥连接的两个物理网段

1.理解“网桥”的含义

也有人把“网桥”比喻成一个聪明的中继器（**Repeater**）。因为中继器只是对所接收的信号进行放大，然后直接发送到另一个端口连接的电缆上，主要用于扩展网络的物理连接范围；而网桥除了可以扩展网络的物理连接范围外，还可以对**MAC**地址进行分区，隔离不同物理网段之间的碰撞（也就是隔离“冲突域”）。集线器和中继器都是物理层设备，而网桥属于二层设备。

我们经常听到这样的说法，那就是“网桥”是一种可连接不同网段的二层网络设备（二层交换机也一样），一个端口可以连接一个网段。所以很多人不总在纳闷，网桥怎么能连接不同网段呢？其实这是因为大家对这里所说的“网段”并不理解。其实这里“网段”更准确地讲应该是叫“物理网段”，是指**IP**地址属于同一网络地址段（也就是**IP**地址中的网络**ID**一样），位于不同地理位置的不同**LAN**分段，是基于物理意义上的地理区域进行划分的。我们常说的网段是指**IP**地址属于不同网络地址段的网络或子网，是一个三层概念，其实这应该叫做逻辑网段，是基于逻辑意义上的网络地址进行划分的。

无论是网桥，还是二层交换机，虽然每个端口可以连接一个网段，但是它们所连接的主机都在同一网络，或者同一子网中。如连接的主机位于不同办公室或者不同办公楼中，则可采用同一网络地址的两个或多个小**LAN**，以组成一个可以统一管理的大**LAN**。但要注意的是，因为网桥只有两个端口，所以所连接的两个物理网段的主机通常

就是由当时的集线器进行集中连接的（网桥端口通常不是直接连接主机的）。软件中通常所说的桥接（如VMware中的桥接工作模式）也就是网桥的作用，它连接的也是同一网络或子网中的两个网段。

2.网桥工作原理解析

前面说到了网桥具有两种主要特性：一是可基于物理网段的MAC地址进行学习，二是可以隔离冲突域。下面通过一个示例来进行解析。

假设图5-34中所示的物理网段1和物理网段2中的主机都是通过集线器集中连接的，则这样这两个物理网段各自形成一个冲突域，因为集线器是采用共享介质传输的，而网桥的背板信道不是共享的（每个端口的数据收发都有一条单独的信道），所以一个集线器就是一个冲突域。网桥的数据转发原理如图5-35所示。下面是具体的解析。

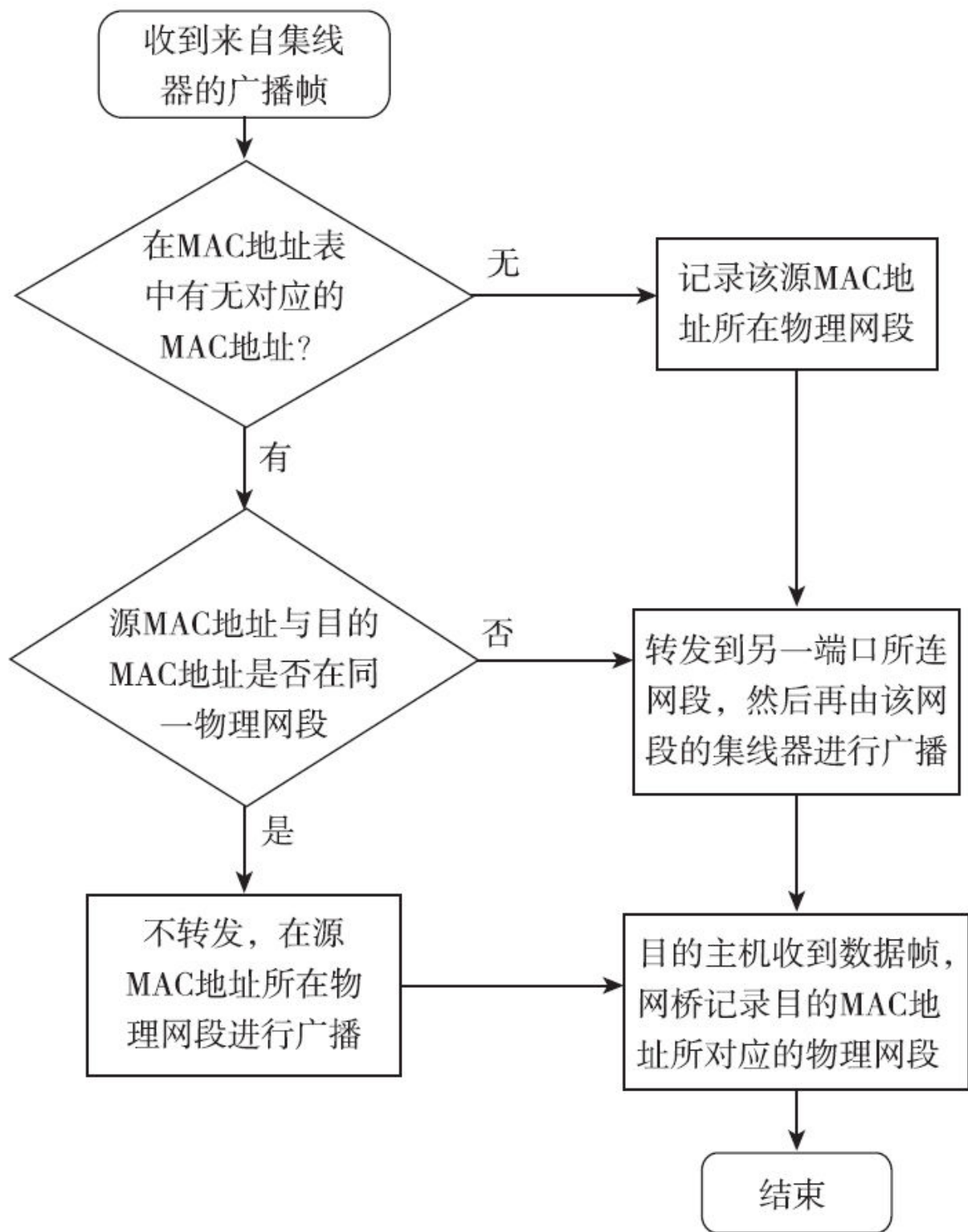


图 5-35 网桥数据转发原理示意图

说明 MAC地址表也就是通常所说的CAM（Content Addressable Memory，内容可寻址存储器）表，保存的是对应MAC地址主机与所连接的交换机端口的映射。这个映射表项可以由管理员手动绑定创建，也可以由交换机自动学习得到。在交换机上可以通过一些命令（如Cisco交换机是使用show mac-address-table命令）查看。下面是一个在交换机上查看MAC地址和端口映射表的示例，其中列出了交换机中为CPU分配的静态（static）MAC地址和通过学习功能自动学习得到的动态（dynamic）MAC地址，其中的Ports列显示的是对应MAC地址主机所连接的端口，VLAN列则为对应主机连接端口所属的VLAN。

```
switch#show mac-address-table
```

Mac Address Table			
Vlan	Mac Address	Type	Ports
All	0100.0ccc.cccc	STATIC	CPU
All	0100.0ccc.cccd	STATIC	CPU
All	ffff.ffff.ffff	STATIC	CPU
1	0000.0c07.accb	DYNAMIC	Gi0/1
1	0002.8501.de00	DYNAMIC	Gi0/1

1	0015.f915.8e80	DYNAMIC	Gi0/1
1	0016.7694.c009	DYNAMIC	Gi0/1
1	0020.ed14.399c	DYNAMIC	Gi0/1
1	0030.b637.8e10	DYNAMIC	Gi0/1
1	0050.ba10.404a	DYNAMIC	Gi0/1
100	0007.847b.c40a	DYNAMIC	Gi0/1
100	00d0.d3a4.7cec	DYNAMIC	Gi0/1
110	0006.28bb.71c0	DYNAMIC	Gi0/1
110	00d0.d3a4.7cec	DYNAMIC	Gi0/1
120	0000.b497.8250	DYNAMIC	Fa0/20
120	0002.b3d8.68e7	DYNAMIC	Fa0/20
120	0002.b3d8.6928	DYNAMIC	Fa0/20
120	0003.a03a.03fc	DYNAMIC	Fa0/19

现假设图5-34所示网络中的一台PC要向另一台PC发送数据。因为集线器也是物理层设备，不能识别帧中的MAC地址，所以无论是哪台主机要发送数据，在集线器上都是以广播方式进行的，连接该集线器上的所有节点都会收到这个广播帧，包括网桥连接到该集线器的端口。

1) 当网桥收到集线器的广播帧后，网桥会把帧中的源MAC地址和目的MAC地址与网桥缓存中保存的MAC地址表进行比较。

2) 最初，网桥的缓存中是没有任何MAC地址的，所以一开始它也不知道哪台主机在哪个物理网段上，收到的所有帧都直接以泛洪方式（也是复制原数据帧）转发到另一个端口上，同时会把数据帧中的源MAC地址所对应的物理网段记录下来（其实就是与对应的网桥端口对应起来）。

3) 在数据帧被某个PC机接收后，也会把对应目的MAC地址所对应的物理网段记录在缓存中的MAC表中。这样，经过多次这样的记录，就可以在MAC地址表中把整个网络中各主机MAC地址与对应的物理网段全部记录下来。因为网桥的端口通常是连接集线器的，所以一个网桥端口会与多个主机MAC地址进行映射。

4) 当网桥收到的数据帧中源MAC地址和目的MAC地址都在网桥MAC地址表中可以找到时，网桥会比较这两个MAC地址是否属于同一个物理网段。如果是同一物理网段，则网桥不会把该帧转发到下一个端口，直接丢弃，起到冲突域隔离作用。相反，如果两个MAC地址不在同一物理网段，则网桥会把从一个物理网段发来的帧转发到连接另一个物理网段上，然后再通过所连接的集线器进行复制方式的广播。

5.8.3 二层交换机概述

在计算机网络设备还有一种是工作在数据链路层的，那就是二层交换机（其实三层或以上层次交换机同样提供二层交换功能）。

交换机（Switch）可以说同时是集线器和网桥的升级换代产品，因为交换机具有集线器一样的集中连接功能，同时它又具有网桥的数据交换功能。所以可以这样说，交换机是带有交换功能的集线器，或者说交换机是多端口的网桥。外形上，集线器与交换机产品没什么太大区别，如图5-36a所示为一款集线器产品，而图5-36b所示为一款交换机产品。



a) 集线器



b) 交换机

图 5-36 集线器与交换机的外观比较

从图中的对比可以看出，交换机与集线器一样，是一个具有许多同类端口的网络设备。当然图中的对比仅能起到一般意义上的外观比较，实际上，因为交换机发展相当快，其应用向两个不同的方向发展，所以在外观上也有很大的区别。小的桌面交换机就像我们现在用

的Modem一般大（集线器也有这样小的），而大的则采用模块化结构，机箱较大，而不是像图中显示那样像一个长方形盒。图5-37所示代表小型固定端口的桌面型交换机和大型模块化交换机。



图 5-37 小型固定端口交换机与模块化交换机比较

1.交换机的主要特性

交换机（此处特指二层交换机）是上节介绍的网桥的升级产品，同样是工作在网络体系结构中第二层（数据链路层）。但它与网桥相比又具有一些特性，具体体现在以下几个方面：

（1）具有多个交换端口

我们知道网桥通常只是两个交换端口，其设计目的主要就是用来连接两个距离超过单段网线传输限制的物理网段（当然也可以用来直接连接两台主机），所以它的应用受到比较多的限制。再加上当时用于主机和其他网络设备集中连接的设备仍是传输效率和信道利用率都非常低下的集线器，根本不适应于计算机网络的发展。有了交换机

后，一台交换机可以有多个端口，而且与网桥一样，不仅每个端口可以连接一个不同的物理网段（交换机上一个端口对应一个物理网段），还可以有大量的端口来集中连接主机，这时交换机就可以同时担当集线器和网桥的双重角色，而且在使用性能和扩展性能、交换性能等方面都有较大提高，大大促进了计算机网络的发展。

（2）数据转发效率更高

在网桥时代，集中连接主机的仍是集线器，而我们知道集线器发送数据是采用广播方式，所以信道中的无效载荷比例相当高，造成数据转发率和信道利用率都非常低。而有了交换机后，因为大多数主机都是直接连接在交换机端口上，即使不是，也主要是连接在其他交换机端口，所以数据的转发基本上都是通过提取帧中的**MAC**地址直接发送到目的主机上的，而不是通过广播方式（仅在未知目的**MAC**地址时采用广播），数据转发效率和信道利用率都大幅提高。

（3）更强的**MAC**地址自动学习能力

我们知道，网桥通常只有两个端口，仅可以连接两个由集线器集中连接的物理网段，所以它的**MAC**地址自动学习功能仅限于它的两个端口与对应的物理网段的映射。这样就造成了，一个网桥端口要与多个源主机**MAC**地址之间的映射，也就是一对多映射关系。而交换机上的端口多数是直接连接主机的，所以在映射表中基本上都是一个源主

机MAC地址与一个交换端口间的一对一映射。一对一的映射查找起来明显比一对多的映射效率要高，所以交换机在数据转发效率要高于网桥。另外，交换机的缓存通常比网桥的要大，所以交换机中可以保存的MAC地址与端口映射表较多，更适用于较大网络。

2.交换机与集线器的区别

交换机最开始是为了解决集线器共享传输介质、端口带宽过窄、容易产生广播风暴而产生的。最初的交换机是工作在OSI开放体系结构中的第二层，所以又称二层交换机。交换机与集线器的区别主要体现在如下几个方面：

(1) 在OSI中的工作层次不同

交换机（特指二层交换机，下同）和集线器在OSI开放体系模型中对应的层次不一样，集线器工作在第一层（物理层），而交换机至少是工作在第二层，更高级的交换机可以工作在第三层（网络层）、第四层（传输层）和第七层（应用层），对应也就有三层交换机、四层交换机、七层交换机等之说了。本章仅介绍二层交换机。

(2) 数据传输方式不同

集线器的数据传输方式是多次复制方式的广播传输，而交换机的数据传输是有目的的，数据只对目的节点发送，只是在自己的MAC地

址表中找不到的情况下第一次使用以FF-FF-FF-FF-FF-FF作为MAC地址的“泛洪”广播方式传输。所以，交换机在数据传输效率和信道利用率方面要远高于集线器，集线器更容易产生“广播风暴”。

(3) 背板信道占用方式不同

在带宽占用方面，集线器所有端口都是共享集线器背板中的一条信道带宽，而交换机的每个端口的收、发都有独享的背板信道带宽，属于“交换”方式。这样一来更进一步使得交换机的数据传输效率以及传输性能要远高于集线器。

(4) 数据通信方式不同

集线器因为是所有端口共享一条背板信道，所以只能采用半双工方式进行传输，同一时间，要么是接收数据，要么是发送数据。而交换机中各端口的信道是采用矩阵交换方式，可以同时进行数据交换，也就是可以进行全双工数据传输。这也使得交换机比集线器的数据通信效率要远高于集线器。

3.交换机的分类

性能越强的设备，应用越广，这也导致了该设备的技术不断发展，基于各种不同应用的设备类型也越多。交换机就是这样一种应用非常广泛的网络设备，它自最初出现至今，各种不同的交换机类型不

断涌现，令人目不暇接。下面介绍一些目前仍广泛应用的交换机分类方式。

（1）根据网络类型划分

根据交换机所应用的局域网类型可以将局域网交换机分为标准以太网交换机（10Mbps传输速率）、快速以太网交换机（100Mbps传输速率）、千兆以太网交换机（1000Mbps传输速率）、十千兆以太网交换机（10000Mbps传输速率）等。

（2）按交换机结构划分

如果按交换机结构来划分的话，交换机可分为固定端口交换机和模块化交换机两种。固定端口顾名思义就是它所带有的端口是固定的。例如，如果是8端口的，那就最多只能接8个设备，不能再添加；如果是16个端口的也就只能有16个端口，不能再扩展，以此类推。目前这种固定端口的交换机基本上都属较低档次的。模块化交换机上就是交换机上除了有部分固定的端口外，还可通过插入扩展模块，来扩展端口数量、所支持的传输介质/网络协议/业务类型。图5-38a、b所示分别为一款固定端口交换机和一款模块化交换机。



a) 固定端口



b) 模块化

图 5-38 固定端口交换机和模块化交换机示例

(3) 按是否支持网管功能划分

按交换机是否支持网络管理功能可以将交换机划分为网管型和非网管型两大类。网管型交换机可以通过控制端口（**Console**）或**Web**界面进行配置与管理，非网管型交换机则不能进行任何配置与管理，仅可按照出厂的默认设置进行工作，也就是通常所说的傻瓜型交换机。

从外观上基本上可以判断一个交换机是否具有网管功能，因为网管型交换机都有一个**Console**控制端口，一般为**RS—232**串口型的（也有用普通的**RJ—45**接口的，但会有一个**Console**标注的字样）。图 5-39所示为一个带有**RS—232**控制端口的网管型交换机。



图 5-39 网管型交换机

5.8.4 二层交换原理

二层交换机与前面介绍的网桥一样，也是工作在OSI参考模型的数据链路层，可以直接根据帧中的目的MAC地址把数据发送给相应端口上连接的主机。在二层交换机中也有用于数据帧转发的MAC地址与端口的映射表（也就是前面所说的CAM表），列出了哪个MAC地址连接的是哪个端口。当在映射表中没有数据帧中对应的目的MAC地址时才进行“泛洪”（以复制方式在除源端口外的其他所有端口上进行转发）。但是要注意，交换机缓存空间毕竟有限，可以存储的MAC地址和端口映射表项也有限，所以当网络比较大时交换机中的缓存空间就不能保存网络中所有节点MAC地址与交换机端口的映射关系了。

1.CAM表的建立

交换机的这张CAM表可以通过多种方式获得，比如静态配置、动态学习，针对多播还可以通过各种多播协议，比如IGMP嗅探、GMRP协议等方式（注意，多播转发表不能通过学习获得，而且多播转发项跟普通转发项不同，跟其对应的出口不止一个，而是一个出口集合，多播方面的具体知识参见笔者编著的《Cisco/H3C交换机高级配置与管理技术手册》一书）。

在进行数据转发的同时，交换机还有一个学习的过程，它包括两个方面：①交换机在接收到数据帧时会把其中的源MAC地址提取出来，查询CAM表，看CAM表中是否有针对该MAC地址的转发项，如果没有，则把该MAC地址和接收到该MAC地址的端口绑定起来，插入CAM表项，这样当接收到一个发送到该MAC地址的数据帧时，就不需要向所有端口广播，而仅仅向这一个端口发送即可。②在接收到的数据帧中目的MAC地址未知情况下，通过向交换机上其他所有端口进行广播，接收广播帧的节点应答广播帧后，便获知了原来数据帧中对应的目的MAC地址所连接的端口，这时交换机又会把该MAC地址与所连端口的对应表项插入到CAM表中。

注意 数据帧的转发是依据目的MAC地址查询CAM表，而CAM表的学习则是以源MAC地址为依据的。但要注意的是，交换机动态学习的CAM表项并不是一成不变的，而是启动一个定时器，当该定时器递减到零时，该CAM表项被删除。每使用一次该CAM表项进行转发，就恢复定时器初始值。

上述介绍是在没有考虑VLAN（虚拟局域网）的情形下阐述的，现在的交换机一般都实现了VLAN，所以CAM表就有了变化，由原来的两项对应关系（MAC地址与所连接的交换端口）变成了三项对应关系（MAC地址，VLAN ID，交换机端口），这样当接收到一个数据帧的

时候，交换机就要同时根据数据帧的目的MAC地址和VLAN ID两项来查询CAM表，找到接口后把该数据帧转发出去。

如果交换机根据MAC地址和VLAN ID查询CAM表失败，即没有该MAC和VLAN ID的对应关系，则交换机把该数据帧对该VLAN包含的（除接收端口以外的）所有端口进行广播。但如果只根据CAM表来确定一个VLAN包含哪些端口，则必须遍历整个CAM表，这样如果CAM表的规模非常大，则查询的效率非常低，所以一般的交换机上在实现VLAN时，还会创建另外一张表，即VLAN配置表。该表包含了VLAN ID和交换机上所有端口的对应关系，即只要根据VLAN ID查询该表就可以找到该VLAN包含的所有端口，这样在进行VLAN内广播的时候，就非常容易了。

2.二层交换原理

因为交换机有多个端口，可直接连接主机或其他交换机。当数据帧发送到本交换机所连接的主机时，交换机就可以根据帧中目的MAC地址直接把数据从对应的端口上发送到所连接的主机上。如果数据发送到本交换机所连接的其他交换机上的主机时，则本交换机先把该数据帧发送到连接到对应交换机的端口上，然后再由那台交换机根据目的MAC地址从对应端口上发送到目的主机上。

总体来说，二层交换原理与网桥的数据交换原理差不多，具体如图5-40所示。不同的只是现在的交换机端口通常不是连接集线器，而是直接交换机和主机，所以在每个端口所连接的物理网段中也采用数据交换方式，而不采用集线器那样的复制类型的广播方式。下面进行具体的解析。

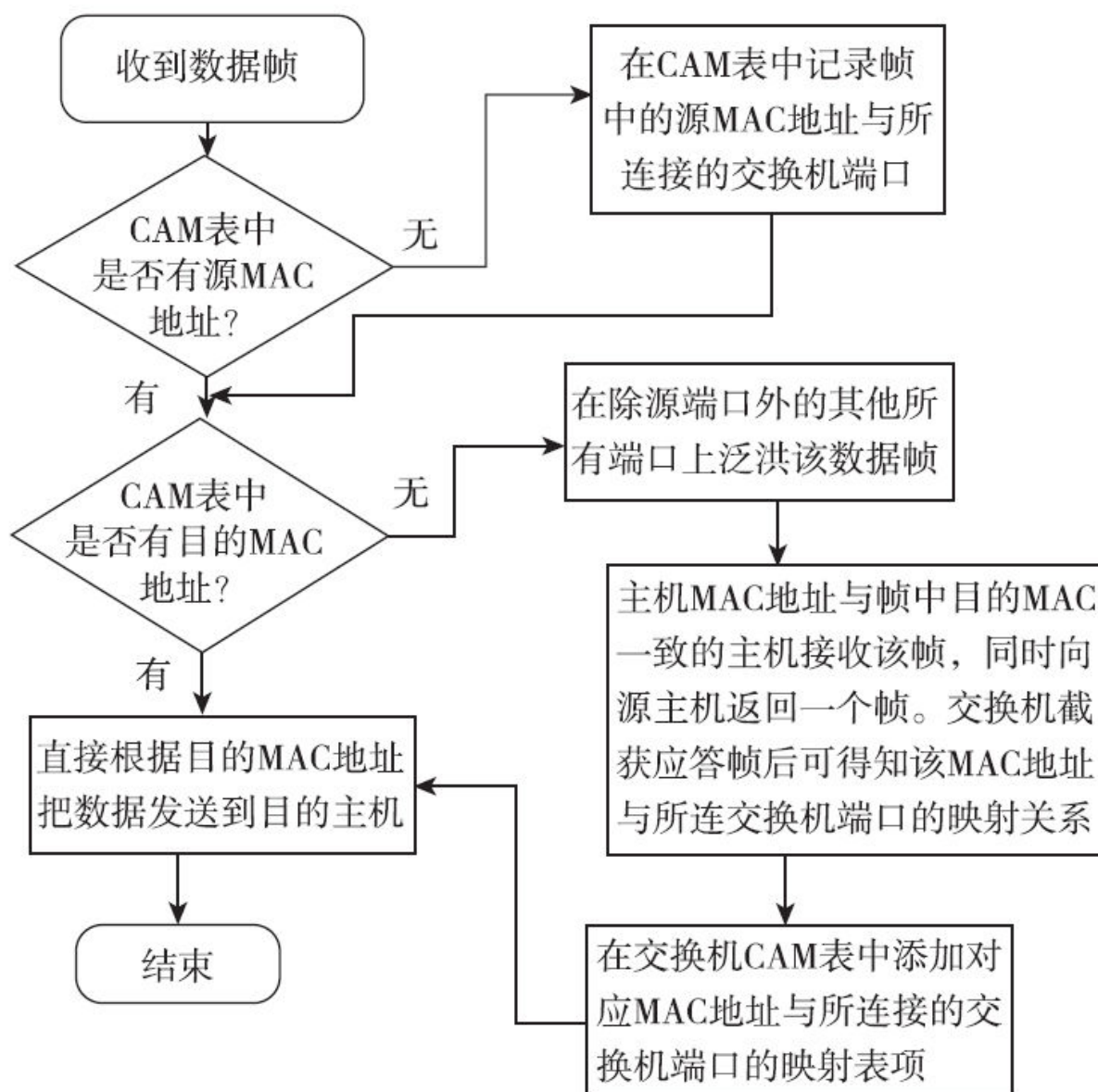


图 5-40 二层交换原理

1) 当交换机从某个端口收到一个数据帧后，先读取帧头部的源MAC地址，并与自己缓存中的映射表（CAM表）进行比较，如果没有找到，则在CAM表中添加一个该源MAC地址与发送该帧的源端口映射表项。这就是交换机的MAC地址自动学习功能。

2) 如果在CAM表项查到了帧中源MAC地址，则继续查看是否有帧中目的MAC地址所对应的映射表项。如果有，则直接把该帧转发到目的MAC地址节点所连接的交换机端口，然后由该端口发送到目的主机。

3) 如果在交换机CAM表中没有找到帧中目的MAC地址所对应的表项，则把该数据帧向除源端口外的其他所有端口上进行泛洪。

4) 当MAC地址与帧中目的MAC地址的主机接收了该数据帧后就会向源主机产生一个应答帧，交换机获取该应答帧后从其中的源MAC地址中获取了对应的MAC地址和所连接端口的映射关系，并添加到CAM表中。这样下次再有MAC地址为这个MAC地址的帧发送时交换机就可以直接从CAM表中找到对应的转发端口，直接转发，不用再泛洪了。

第6章 介质访问控制子层

介质访问控制子层（**MAC**子层）是局域网体系结构中划分的子层，对于广播型网络（如以太网、**WLAN**）是非常重要的，因为它担负了两方面的主要职责：一是如何在局域网内寻址（也就是找到目的节点），二是如何解决多路通信中介质争用的现象。在非广播型的点对点网络（广域网基本上属于点对点网络，但像卫星通信类的网络仍属于广播型网络）中，却不存在这两个问题，因为点对点网络中，目的节点是唯一的，不需要寻址，而且双方之间的通信也只在彼此之间发生，没有其他用户使用这个介质，所以也不存在介质争用现象。也正因如此，本章所介绍的内容主要适用于局域网，包括以太网和**WLAN**无线局域网。

本章首先会分析局域网中的信道类型，然后介绍**MAC**子层的各方面功能和技术实现原理（如**CSMA**、**CSMA/CD**介质访问控制原理），后面主要介绍各以太网体系结构、规范和帧格式封装，主要**IEEE 802**局域网标准（如**IEEE 802.1d**、**IEEE 802.1q**、**IEEE 802.1w**、**IEEE 802.1s**和**IEEE 802.1x**）工作原理，最后综合介绍各主要**WLAN**规范、**CSMA/CA**原理和**WLAN**帧格式封装。

6.1 MAC子层基础

MAC子层是有线局域网体系结构和WLAN体系结构中数据链路层的一个子层。它有两个主要作用，一是用来寻址（这里是MAC地址），也就是寻找目的节点；二是用来解决网络中多个用户争抢共享物理介质或者共享信道的现象。如在总线型局域网中，所有或者多个用户计算机连接在一条总线上，不同用户计算机间的通信肯定会存在介质争用现象。

经验之谈 大家千万不要认为仅有完全的总线型拓扑结构网络才有介质争用现象发生，其实只要网络存在“总线”部分，也就是只要网络中存在可能有多个用户通过同一条介质访问同一个节点，就存在介质争用。即使在最简单的星型结构单元以太网中，也会存在介质争用现象，因为每个用户计算机与交换机的连接都是通过一条电缆进行的，网络中可能有多个其他用户要同时访问该用户。在企业网络中主要使用的树型结构以太网中，这种介质争用现象就更普遍了，因为不同交换机之间的连接也通常是通过一条级连（Uplink）电缆进行的，这样两个交换机之间用户的访问就都必须通过这一条级连电缆，介质争用就更不可避免了。有关这几种拓扑结构连接原理，大家可以参见第1章的相关内容。

在Internet的TCP/IP协议体系结构中没有MAC子层，因为在TCP/IP协议体系结构中主要描述的是广域网中的“通信子网”部分，在这部分主要设备就是各种广域网路由器（一般没有传统的以太网交换机），

而在这些广域网路由器之间的连接基本上都是点对点连接的，不存在寻址，也不存在介质争用现象。

6.1.1 两种信道类型

说到MAC子层就不得不提到信道这个概念，因为MAC子层的主要功能之一就是为了解决不同用户通信信道争用问题。但是，并不是所有网络都存在信道争用问题，只是在广播类型网络中存在。我们在前面就介绍了，信道可分为点对点信道和广播信道两大类，对应点对点和广播两种链路，当然也对应点对点和广播这两种网络类型。

所谓点对点信道是指由两个没有经过任何中间设备（也就是两设备的接口是“背靠背”连接的）的节点（也就是通常所见的计算机网卡，或其他网络设备端口）间构成的信道，如图6-1所示。但这还需要看链路上封装的数据链路层协议而定，点对点信道是要封装点对点类型的数据链路层协议（如PPP和PPPoE等），所构成的链路就是我们通常所说的点对点链路。

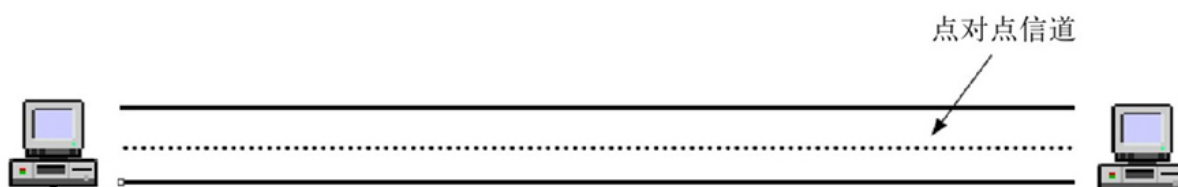


图 6-1 点对点信道示例

因为点对点链路两端的节点只与对方节点进行连接，所以不存在寻址问题。如图6-2所示的网络中，计算机A可以（注意，不是必须，因为具体还要看所封装的数据链路层协议）与计算机B、计算机C、计算机E建立点对点连接，但却不能与计算机D建立点对点连接，因为计算机A与计算机D之间建立连接时必须经过一个中间设备（计算机B，或者计算机C，或者计算机E），就不属于点对点连接了。计算机E却可以与其中的任何一台计算机建立点对点连接。注意，本示例中，每台计算机与另一台计算机的连接都是通过一块单独网卡进行的，这里的点对点链路是在两台计算机上安装的两块计算机网卡之间建立的。

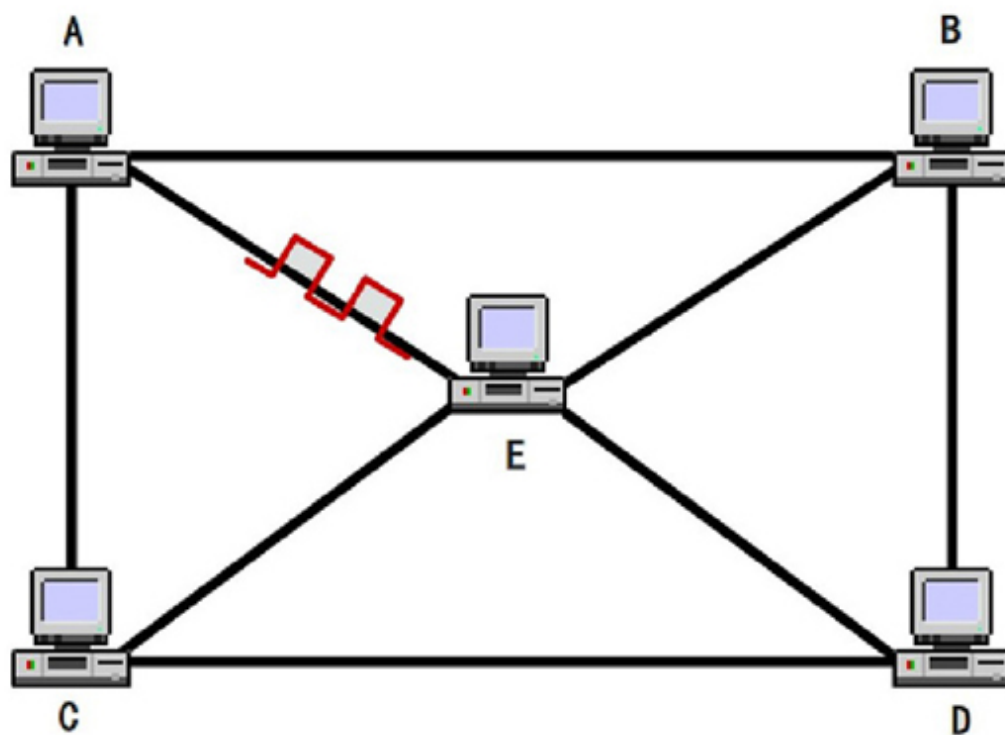


图 6-2 点对点连接示例

广播信道表现为一个信道被多条链路所共享（也就是存在共享信道），也就是存在物理介质共享的情况，如图6-3所示。在广播信道中，一个节点发送的数据可以同时被多个节点接收到，对应的链路就是“广播链路”。当然这里也要根据对应链路上所封装的数据链路协议而定，最典型的广播型数据链路层协议就是我们经常用的以太网协议和WLAN协议。在这些广播网络中，一个用户发送的一个广播包可以通过交换机广播到本局域网中的所有节点上。图6-4所示是在一个总线型网络中，数据在广播信道中同时向多个节点传输的示例。

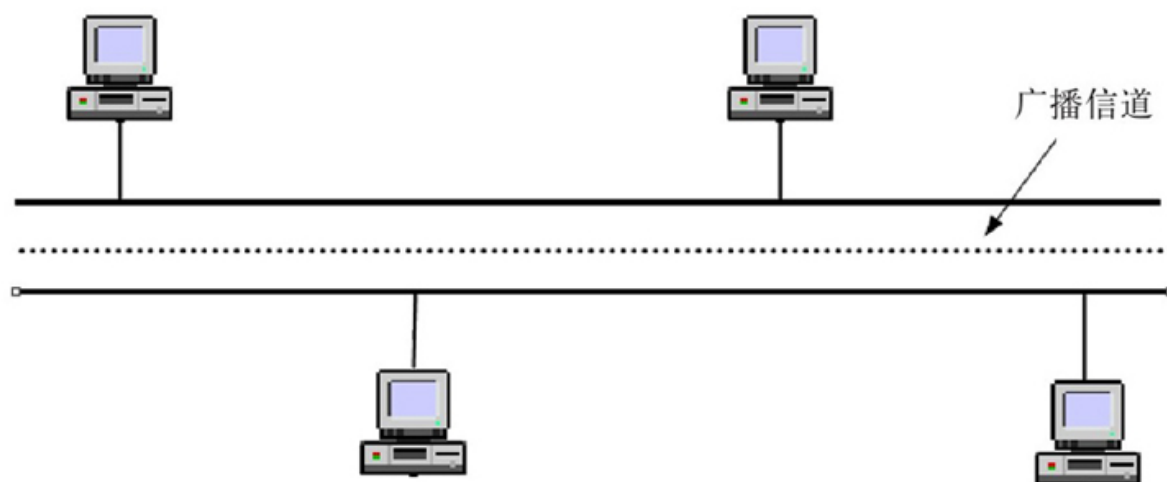


图 6-3 广播信道示例

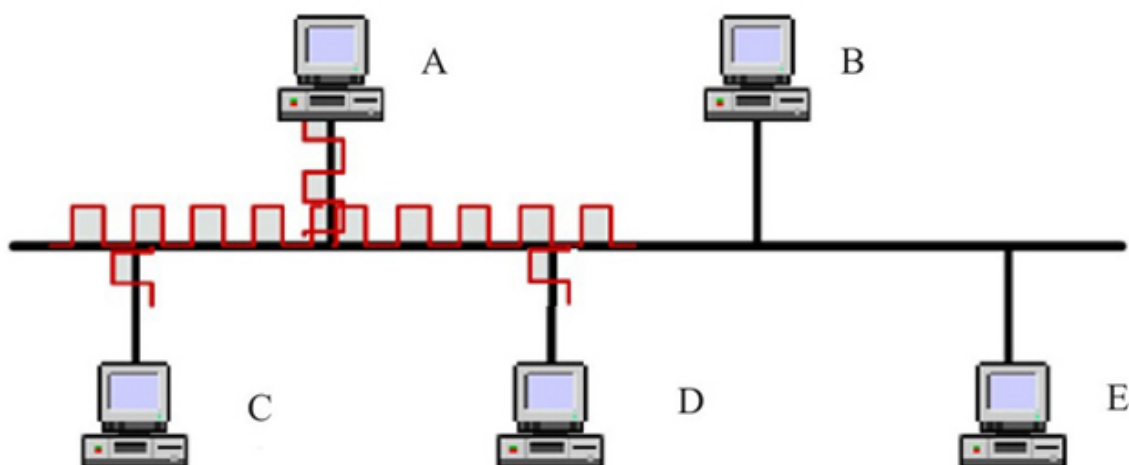


图 6-4 广播数据传输示例

从以上分析可以看出，**MAC**子层仅在广播型网络（如以太网、**WLAN**，以及其他无线网络）中 useful，而对于点对点型网络中就没有存在的意义了，因为在点对点网络中不需要寻址，也不存在一般意义上的信道争用问题（即使存在，也基本上可通过各种信道复用技术来解决）。

6.1.2 MAC子层概述

局域网（包括令牌网、以太网和WLAN等）中与接入的各种传输介质相关的问题都放在MAC子层来解决，而且MAC子层还负责在物理层的基础上实现无差错的通信。具体说，MAC子层的主要功能包括：MAC帧的封装与拆卸（这方面已在第5章有详细介绍了），实现和维护各种MAC协议（这是本章重点之一），比特流差错检测（这方面也已在第5章有详细介绍了），MAC寻址（这也是本章重点之一）等。

因为数据链路层分成了LLC子层和MAC子层（LLC子层与上面的网络层连接，而MAC子层与下面的物理层连接，具体参见图5-3），所以数据链路层有两种不同的数据帧——LLC帧和MAC帧。不过，我们通常所说的“帧”一般是指MAC帧，而不是LLC帧。因为从上层来的数据包，进入到LLC子层后加上LLC子层的协议头和协议尾就形成了LLC帧，然后需要继续向下传输，到达MAC子层后同样再要加上MAC子层的协议头和协议尾，又要进行重新封装，最终形成MAC帧传输到物理层。对于物理层来说，最终的数据链路层帧就是MAC帧。

由于IEEE 802系列局域网标准中规定了不同的MAC子层协议（对应不同的网络标准），所以其MAC帧的帧格式也各不相同（具体将在

本章后面介绍），但不管是哪一种**MAC**协议，都具有**MAC**地址，以便在局域网内部实现二层寻址。

经验之谈 在同一网段的局域网内部的网络是不需要通过三层的**IP**地址来寻址的，可直接通过二层**MAC**地址。但要注意的是这里所说的二层寻址仅用于二层网络设备之间的网络通信，不能作为用户网络应用通信的寻址，因为来自应用层的数据在传输到达对方应用层时，最终依靠的不是这里所说的**MAC**地址，而是网络层的**IP**地址、传输层的端口和应用层的用户进程。无论是在局域网内部，还是在不同网络之间的数据通信都是如此。在一个局域网内部，网络层的功能是由安装在用户主机中的网络操作系统提供的（因为在一个局域网内部没有路由器设备），由主机对来自应用层的用户数据进行三层封装。千万不要认为，在局域网内部进行网络应用就不需要网络层及以上各层。只是纯二层的网络通信（仅用于构建数据通信所需的数据链路，不参与用户应用数据的处理）不需要网络层及以上各层，要进行具体的网络应用仍然需要网络层及以上各层的功能。

在局域网中，**MAC**地址的作用就是用来找到我们要进行通信的计算机，网卡从网上每收到一个**MAC**帧，首先检查其**MAC**地址，如果是发往本节点的帧就收下，然后进行其他处理。但**MAC**地址只能在数据链路层识别，在三层（网络层）上是不能识别的，必须依靠**ARP**映射表来查找对应的目的**MAC**地址。**MAC**帧有以下三种：

□单播帧——目的MAC地址是一个单播MAC地址的帧；

□广播帧——目的MAC地址是一个广播MAC地址（全“1”地址）的帧；

□多播帧——目的MAC地址是一个多播MAC地址（多MAC地址有又许多种，具体参见笔者编著的《Cisco/H3C交换机高级配置与管理技术手册》一书）的帧。

6.1.3 介质争用综述

在6.1.1节说了，广播型网络中就存在介质争用现象，其实这就涉及一个概念——冲突域。什么是冲突域呢？其实也就是可能发生介质访问冲突的节点范围。这时我们要分析两种在局域网中经常用到的设备——集线器（Hub）和交换机（Switch），通过对它们的数据传输原理介绍来帮助理解冲突域。

1.理解冲突域

首先来看一下Hub，在它中间的背板是以总线方式连接各个端口的，如图6-5所示。从图中可以看，在Hub中连接的所有用户都是通过共享一个背板信道进行访问的，所以连接的所有用户在同时访问时都可能发生冲突，所以我们说一个集线器就是一个冲突域。

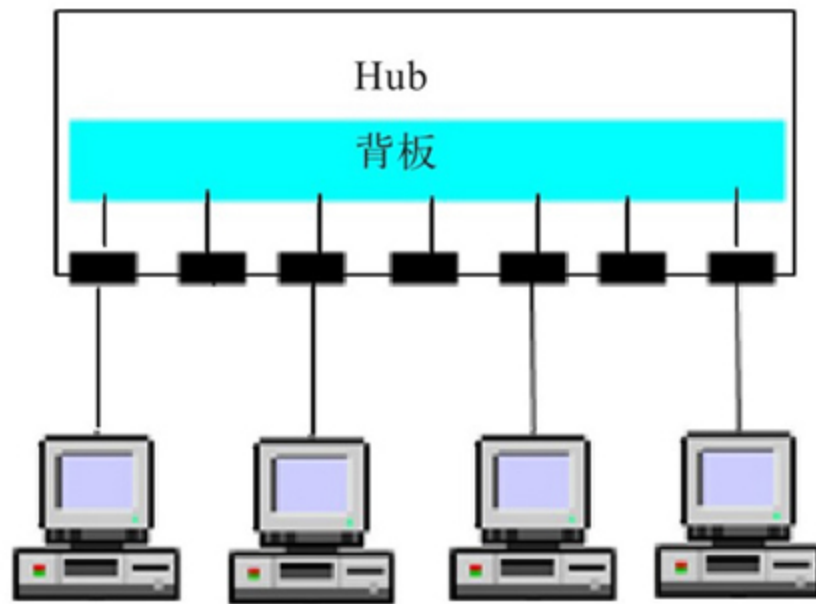


图 6-5 Hub端口与背板的总线连接示意

交换机（Switch）的背板与端口的连接方式与Hub的不一样，在背板中有一个交换矩阵，如图6-6所示。通过交换矩阵，就可以实现任何两个端口间的通信都有一条专用的通道，不同节点间的通信不存在介质争用现象。也正因如此，在交换机中的冲突域就不再是整个交换机了，而是缩小到一个具体的端口了，因为一个端口连接一条电缆，而同时可能有多个用户要与这个端口上的用户进行通信，这时就有可能发生介质访问冲突了。

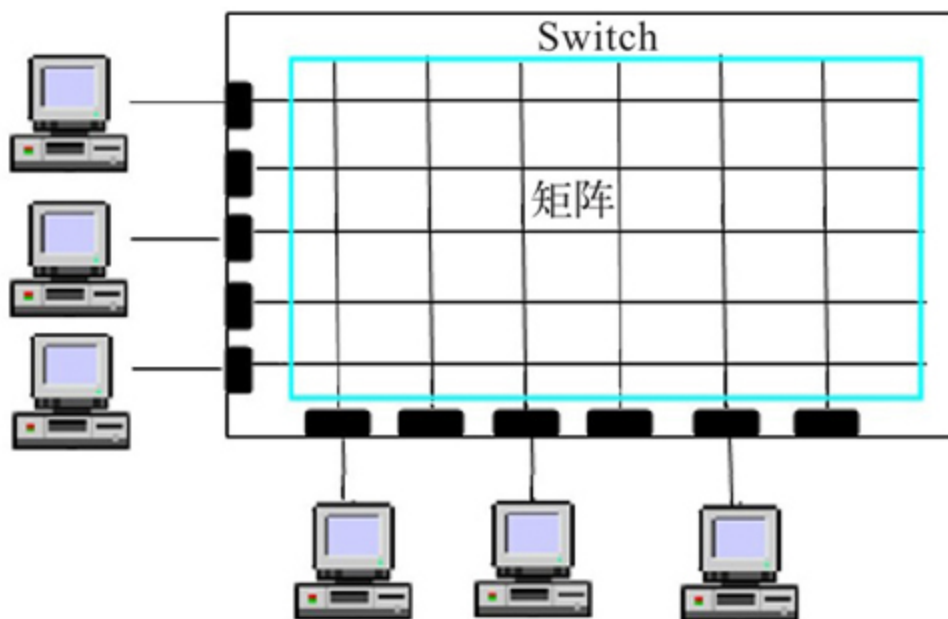


图 6-6 Switch端口与背板板矩阵连接示意图

打个比方来说，在一条十字形的公路上，东、西、南、北四个方向的车是不可能同时通过的，得通过交通信号指示灯来进行指引，每一段时间仅允许相对的两个方向（在有两个车道以上的情况下）的车通过。因为无论是哪个方向的车通过都要经过路中央的那个地方（相当于集线器的背板信道）。但是如果建个立交桥，甚至多层的立交桥，并规定不同方向的车走不同的桥（相当于交换机中的背板交换矩阵），则任何方向都可以在同一时间通车了。

另外，冲突域是有大小的，其大小就是可能共享同一介质的节点数多少。如在Hub中，端口数越多，则共享同一背板信道的用户数越多，冲突域就越大，发生介质争用的可能性就越高，可能造成的冲突

也越大。当然，冲突的大小还与集线器或交换机背板带宽的大小有关，背板带宽越高，数据通过的速率就越高，发生冲突的可能性就越低，冲突就越小。就像同时过一座桥的人越多，发生拥挤的可能也就越高，但如果桥面越宽，允许同时过这个桥的人数就越多，发生拥挤的可能也就越低。

2.介质争用解决方法

既然存在介质争用现象，自然就得想办法来解决，否则冲突一多，自然会导致数据的丢失，或者畸变。就像一群人要同时过一座桥一样，如果人数过多，自然就会有人挤人的现象，甚至有人被挤掉到桥下的危险。但是，不同网络，所采用的共享介质争用解决方法也不一样。

我们在学习前面**IBM**令牌环网络和令牌总线网络时就说过，在这两种看似都是采用共享介质的令牌网络中却不存在介质争用现象，原因就是它们采用了“令牌”控制原理。就是在这两种网络中，用户必须得到网络中唯一的“令牌”才能发送数据。显然同一时间只有一个用户节点可以得到这个令牌，避免了介质争用现象的发生，尽管网络中各用户是共享这一条介质的。就像许多人要从两个方向过一座独木桥，如果桥边有管理员，并规定每个人过桥时都必须先征得管理员同意才能过桥（也就是等一人过完后另一个人才能过桥），这样就不可能发生拥挤现象了，尽管这条桥很窄，一次仅允许一人通过。

那么其他存在介质访问冲突的网络中又是如何解决介质争用现象呢？在广域网中，通常是采用各种对应的信道复用技术来解决，如我们前面介绍的**FDM**信道复用技术就是把一条高带宽的信道划分成带宽较小的多个信道，这样就可以同时进行多路通信了，就像在一条公路上划分多个小车道一样，可以同时通过多部小车一样；**TDM**信道复用技术是通过把不同路通信的数据包分配在不同时隙进行传输来实现信道复用的。

在我们常用的以太网和**WLAN**无线局域网的介质争用解决方案主要有：**CSMA**、**CSMA/CD**和**CSMA/CA**协议，后两者是前者的改进版本，在此先介绍在以太网中使用的**CSMA**和**CSMA/CD**，适用于**WLAN**的**CSMA/CA**将在本章后面介绍。

6.2 CSMA介质访问控制原理

在总线型网络中，每个站点都能独立地决定帧的发送（没有主站点和从站点之分），很显然，如果两个或多个站点同时向总线上发送帧，就会产生介质访问冲突（仅指在没采用信道复用情况下），导致所发送的帧出错。因此，在这种总线型网络中，一个用户数据发送的成功与否，很大程度上取决于发送数据时是否会与其他用户发送的数据产生总线介质争用。这时就需要一种能有效避让冲突发生的技术，确保每个站点在向总线上发送数据时，其他站点均不发送数据，也就是如何使各个站点能尽快地检测到总线介质是否空闲。本节所介绍的CSMA（载波侦听多路访问）就是这样一种能比较有效解决总线型网络中介质争用的技术。

CSMA技术又称LBT（Listen Before Talk，先听后说），也就是先侦听要访问的介质，当发现介质忙时先避让一段时间，不发送数据，仅当侦听到介质空闲时才进行数据发送。在这里涉及一个问题，就是在站点侦听到当前信道中有数据在传输时，要避让多长时间才再次侦听，这就是CSMA技术的退避算法。但CSMA可以采用的退避算法有几种，也可算是对应类型的CSMA，那就是：非-坚持CSMA（non-persistent CSMA）、1-坚持CSMA（1-persistent CSMA）、P-坚持CSMA（P-persistent CSMA）。下面对这些算法进行具体介绍。

6.2.1 非-坚持算法

非-坚持中的“非”的意思就是指各站点不连续侦听总线介质是否空闲，即在发现介质忙时，先停止侦听，等过一段时间再来侦听。具体来讲，非-坚持CSMA就是某站点发现总线介质上没有数据在传输时（处于闲状态），立即发送在等待发送的数据；如果检测到总线介质上有数据在传输时（处于忙状态），则等待一段时间再来侦听，再忙时再等待一段时间来侦听，以此类推，直到发现总线介质处于空闲时本站点才可发送数据。

图6-7所示的就是一个非-坚持CSMA介质访问控制示例。示例中某站点准备要发送一个数据帧Fi。它第一次检测时发现总线是处于忙状态，于是隔了一段时间 t_1 后，再来侦听，发现总线介质还是处于忙状态，于是它再隔一段时间 t_2 来侦听，终于在第三次侦听时发现总线介质是处于闲状态，于是这个站点立即把Fi帧发送出去。

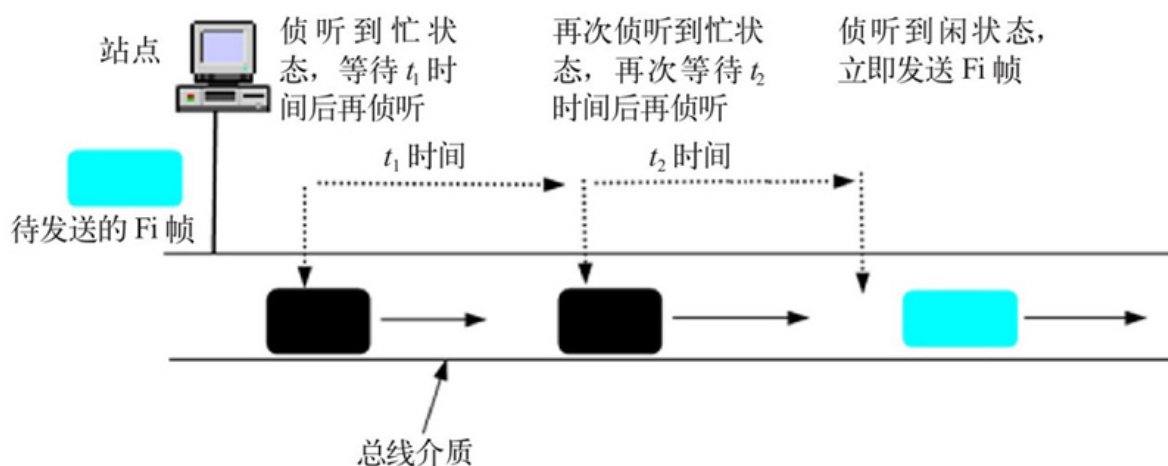


图 6-7 非-坚持CSMA介质访问控制示例

这时大家可能已经有疑问了，那就是这个非-坚持CSMA算法中，在上次侦听发现介质处于忙状态时需要等待多长时间再来侦听介质的状态呢？答案是不固定的，这个等待的时间是根据非-坚持CSMA算法随机产生的（具体产生机制大家可不必深究）。如果采用TDM（时分复用）传输方式，则这个延迟时间就是一个时隙长度。

虽然非-坚持CSMA算法可以在一定程度上减少冲突的发生，但还是不能完全消除冲突的发生，因为毕竟每个站点发送的数据速度不一样，数据在信道中传输是需要时间的，而且等待的时间也不一定就可以满足一个帧完全传输完毕，所以尽管每个站点都是在检测到介质空闲时才发送数据，但仍可能存在前一个帧还在信道中传输时，后一个站点的帧就追上来了，进而发生冲突。

另外，非-坚持CSMA算法还存在一个明显的缺点，这就是一旦侦听到介质忙就马上延迟一个随机时间再重新侦听，但很可能在再次侦听之前信道就已经空闲了。也就是非-坚持CSMA不能把信道刚一变成空闲的点找出，影响了信道利用率的提高。所以这种算法主要适用于小型的总线，或者树型拓扑结构网络中，不适用于像现在大型的树型结构以太网中。为了克服这一缺点，可采用下面介绍的两种坚持CSMA。

6.2.2 1-坚持算法

1-坚持中的“1”有两层含义：一是指发现总线介质忙时一直持续不间断侦听，直到发现介质处于闲状态；二是在侦听到介质处于空闲状态后一定（也就是100%）发送数据。前面介绍的非-坚持CSMA算法是在发现介质忙后，即随机等待一个延时，然后继续侦听，发现介质处于闲状态后立即发送数据；而此处的1-坚持CSMA算法中是在发现介质忙时不等待，继续侦听，一旦发现空闲即立即发送数据；在数据传送过程中发生冲突时放弃当前的数据传送任务，等待一个延时后再继续侦听。

图6-8所示的是一个1-坚持CSMA介质访问控制示例。示例中某站点准备要发送一个数据帧 F_i 。它第一次检测时发现总线是处于忙状态，于是继续侦听（不等待），第二次侦听发现总线介质还是处于忙状态，于是它再继续侦听，一直到第 n 次侦听时才发现总线介质是处于闲状态，于是这个站点立即把 F_i 帧发送出去。

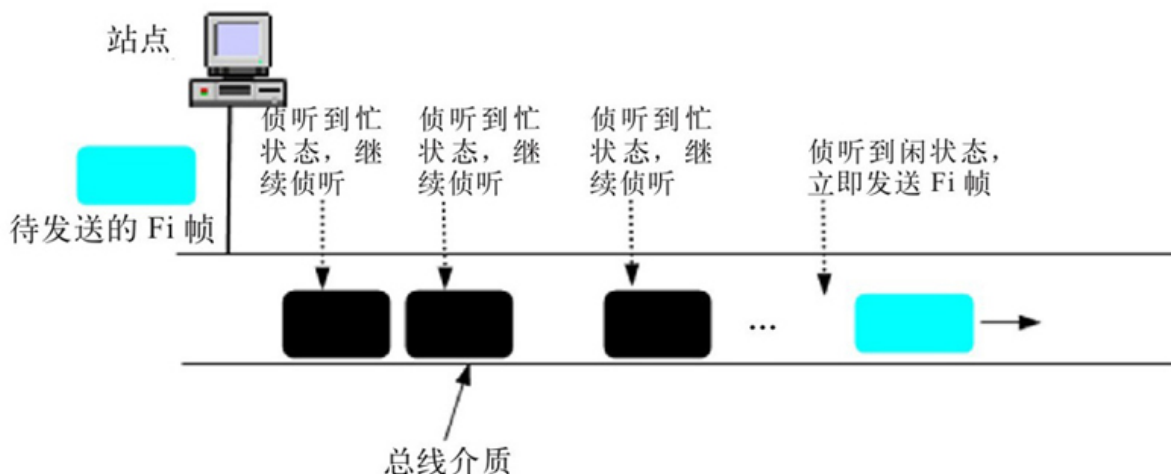


图 6-8 1-坚持CSMA介质访问控制应用示例

很明显，这种算法相对前面介绍的非-坚持算法来说的优点就是提高了介质的利用率，每个站点都在抓住一切有利时机发送数据，因为在没有发生冲突时是无须等待一个随机时延就立即进行继续侦听，不会出现介质处于空闲状态仍没有站点发送数据的情况。但是，该算法仍有两个致命的弱点：一是在有多个站点发送数据的情况下，这种毫不等待的算法，也就使得冲突时常发生。原因就是前面所说的，可能在网络中同时有多个站点在同一时间检测到介质空闲（因为中间没有一个延迟，也就是一直在侦听介质状态），而立即进行了数据发送，所以更容易发生冲突。二是这种算法发现在介质忙时一直侦听，占用了大量网络和设备资源。所以1-坚持CSMA算法也仅适用于小型的总线型或者树型拓扑结构网络，不适用于现在大型的树型结构以太网。

6.2.3 P-坚持算法

既然前面介绍的两种算法都存在明显的不足，自然就会有人继续后面的开发，于是就生产了新的P-坚持CSMA退避算法。其实，P-坚持CSMA算法是前面介绍的非-坚持CSMA和1-坚持CSMA这两种算法的一种折中算法，取两者的长处，而尽量克服了两者的不足。

理解P-坚持CSMA退避算法的关键就是理解其中的P，其是指侦听到介质空闲时发送数据的概率（小于1，也就是小于100%），就是指站点在发现介质空闲时可以立即发送数据的概率为P，也就是不一定（不是100%）发送数据，还有（1-P）概率是不发送数据。其目的就是为了避免与其他站点发生冲突。

具体来讲，P-坚持CSMA算法的具体介质访问控制方法与前面介绍的1-坚持CSMA类似，也是在侦听到介质处于忙状态时持续侦听，当侦听到介质处于空闲状态时，此时站点却不一定马上发送数据，根据概率P（这个P值是算法事先确定好的）来选择发送数据，而在另一个（1-P）概率的时候，即使介质处于空闲状态，也会延迟一段时间t（这个t是指端到端的传播延时）后再重新侦听介质状态。

上面所说的传播延时是有计算公式的，具体如下：

传播延时(μs)=两站点间的距离(m) \div 信号传播速度(m/s)

如已知传播距离为1000km，信号在介质上的传播速率为 2×10^8 m/s，则可计算出它的传播延时为 $1000 \times 1000 \text{m} / (2 \times 10^8 \text{ m/s}) = 0.005 \text{s}$ 。

这里最难理解就是P概率了。打个比方，P为0.8，也就是80%的空闲状态下是可以立即发送数据的，还有20%（1-P）的概率是不发送数据，要延迟一个t时间（也就是传播延时）再来侦听介质状态。如果在某段时间中，前面已有一个空闲状态下没有发送数据，则下一个空闲状态下就很可能立即发送数据了（毕竟有80%的可能性是要发送数据的），否则如果前面连续几个空闲状态下都立即发送了数据，则下一个空闲状态下就可能不发送数据了。总体来说，是否发送数据是随机的，没有固定哪个空闲状态下必须发送数据，或者不发送数据，但整体通信过程中在空闲状态下发送数据的概率都是在P以内。

图6-9所示的是一个P-坚持CSMA介质访问控制示例。示例中某站点准备要发送一个数据帧Fi。它第一次检测时发现总线是处于忙状态，于是继续侦听（不等待），第二次侦听发现总线介质还是处于忙状态，于是它再继续侦听，第3次侦听时发现总线介质是处于闲状态，但根据概率计算出现在不能发送数据，于是等待一个t时间后，再来侦听介质状态，到第4次侦听时发现介质还是处于空闲状态，根据概率计算得出马上可以发送数据了，于是把Fi帧发送出去。

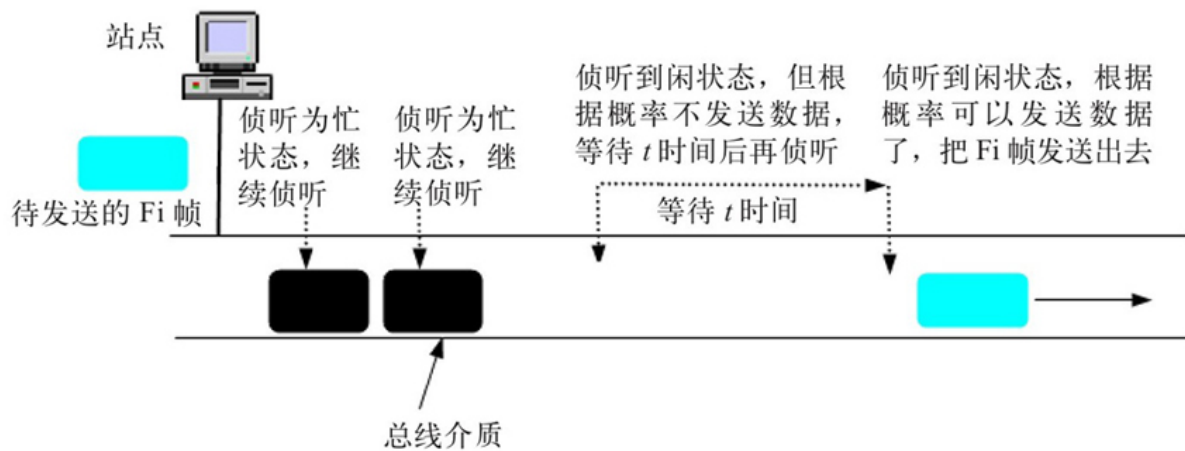


图 6-9 P-坚持CSMA介质访问控制应用示例

从以上分析可以看出，P-坚持退避算法是一种既能像非-坚持CSMA算法那样减少冲突，又能像1-坚持CSMA算法那样减少介质空闲时间的折中方案，也就是综合了前面所介绍的两种算法的优点，以实现优缺点互补。

6.3 CSMA/CD介质访问控制原理

在上节介绍的CSMA介质访问控制中，由于信道传播延时的存在，即使通信双方的站点都没有侦听到载波信号，在发送数据时仍可能会发生冲突，因为他们可能会在检测到介质空闲时同时发送数据，致使冲突发生。尽管CSMA可以发现冲突，但它并没有先知的冲突检测和阻止功能。而本节要介绍的CSMA/CD（Carrier Sense Multiple Access with Collision Detection，载波侦听多路访问/冲突检测）技术却具有冲突检测和阻止功能，它是CSMA技术的改进版本。

CSMA/CD是标准以太网、快速以太网和千兆以太网中统一采用的介质争用处理协议（但在万兆以太网中，由于采用的是全双工通信，所以不再采用这一协议）。CSMA/CD的最大亮点就是它在侦听到有冲突发生时（同时有多路数据发送时，信号强度肯定不一样），可以立即中止数据帧的发送，快速地终止被破坏的帧可以节省时间和带宽；并且发送一个阻塞信号，以强化冲突，使其他站点更容易检测到有冲突的发生，不再同时发送数据了，可避免更多的帧发生冲突。

6.3.1 CSMA/CD原理综述

CSMA/CD的介质访问控制原理包含四个处理内容：侦听、发送、检测、冲突处理，可以用几句话来概括：

□先听后说（“听”是指侦听，“说”是指发送数据，下同），边听边说；

□一旦冲突，立即停说；

□等待时机，然后再说。

具体解释如下（可参见图6-10所示的示例）：

1) 当一个站点想要发送数据的时候，它首先要检测总线介质上是否有其他站点正在传输，即侦听介质是否空闲（也就是前面所说的“先听”）。

2) 如果信道忙，则继续侦听，直到侦听到介质状态为空闲；如果侦听到介质状态为空闲，站点就准备好要发送的数据（也就是前面所说的“后说”）。

3) 在发送数据的同时，站点继续侦听总线介质（也就是前面所说的“边听边说”），确信没有其他站点在同时传输数据才继续传输数据。因为有可能两个或多个站点都同时检测到介质空闲，然后几乎在同一时刻开始传输数据。如果两个或多个站点同时发送数据，就会产生冲突。若无冲突则继续发送直到发完全部数据。

4) 若检测到有冲突，则立即停止发送数据（也就是前面所说的“一旦冲突，立即停说”），同时发送一个用于加强冲突的JAM（阻塞）信号，以便使网络上所有站点都知道网上发生了冲突，不再接收原来的帧，转而接收这个阻塞帧。本站点然后等待一个预定的随机时间（也就是前面所说的“等待时机”），且在总线为空闲时，再重新发送数据（也就是前面所说的“然后再说”）。

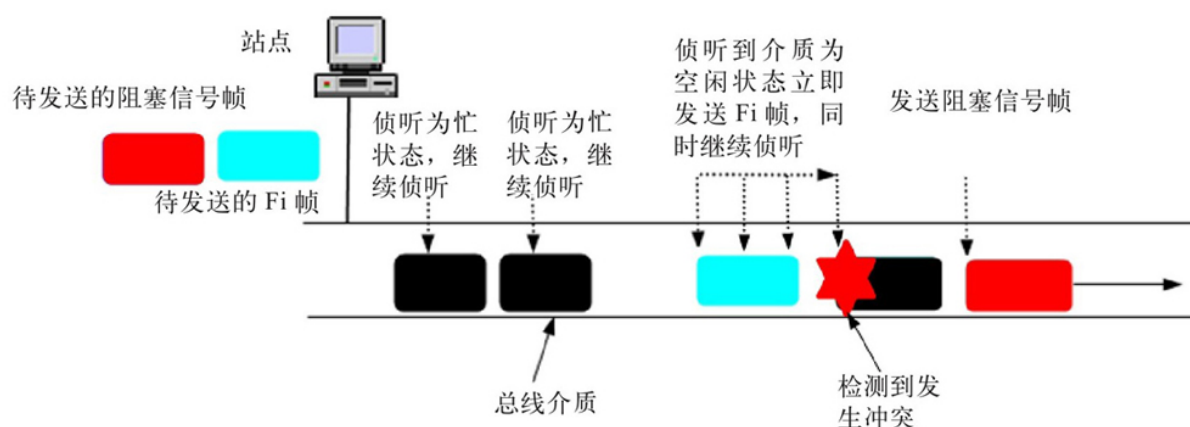


图 6-10 CSMA/CD介质访问控制应用示例

有人将CSMA/CD的工作原理形象地比喻成很多人在一间黑屋子中举行讨论会，参加会议的人都是只能听到其他人的声音，看不到人。会议中规定，每个人在说话前必须先倾听，只有等会场安静下来后，他才能够发言。这时，可将人们在发言前进行的倾听（以确定是否已有人在发言）的动作比喻为CSMA/CD的载波侦听；将在会场安静的情况下每人都有平等机会讲话比喻为CSMA/CD的多路访问。如果有两人或两人以上同时说话，大家就无法听清其中任何一人的发言，这种情

况称为发生冲突；发言人在发言过程中要及时发现是否发生冲突，这个动作称为冲突检测；如果发言人发现冲突已经发生，这时他需要停止讲话，然后随机后退延迟，再次重复上述过程，直至讲话成功。如果失败次数太多，他也许就会放弃这次发言的想法。

6.3.2 冲突检测原理

为了确保所有站点在发送数据的站点发送完数据前开始接收数据，以太网定了一个最小帧（这个最小帧的有效载荷不能少于46字节），最小帧的大小与网络分布的距离、传输介质类型和到达目的节点前所使用的中继器数量等有关。综合这些因素专家们定义了一个公认值——以太网时隙，为10Mbps速率下传输512位数据所用的时间，即51.2 μ s。

当有两个或多个正在传输数据的站点检测到它们发送的数据发生了冲突时，它们都会通过发送一个阻塞帧（它是一个32位全为1的帧）来进行响应。下面以一个示例来描述CSMA/CD的冲突检测原理。

1) 一开始（ $t=0$ 时），站点A在介质空闲时发送一个帧，如图6-11所示。

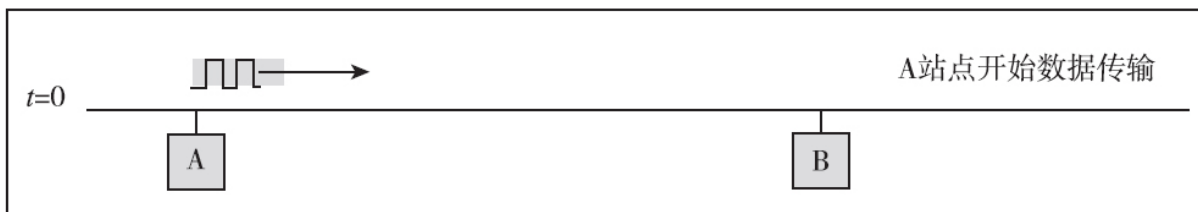


图 6-11 站点A开始发送数据

2) 一段时间后, 在站点A传输的数据帧还没到达站点B前, 站点B也可能检测到介质为空闲的, 于是也开始进行数据传输, 如图6-12所示。

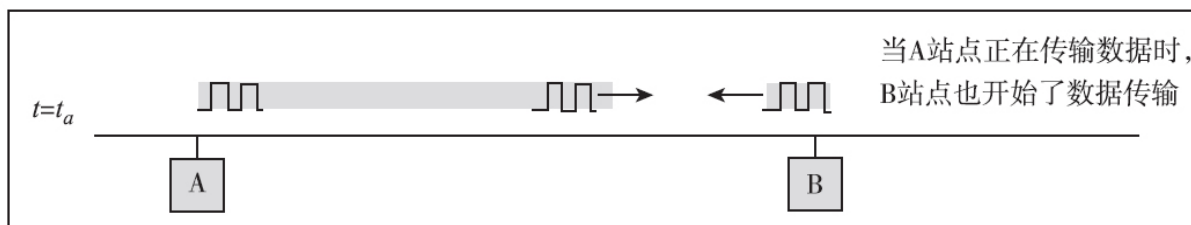


图 6-12 站点B也开始数据传输

3) 从A站点发送数据开始, 过一段时间后 (相当于站点A到达站点B的传播延时 t_p) , 站点B检测到了来自站点A的数据, 检测到了发生了冲突, 立即中止原来的数据传输, 发送一个32位的阻塞信号帧。但是站点A此时仍还不知道站点B也在发送数据, 仍不知道有冲突发生, 如图6-13所示。

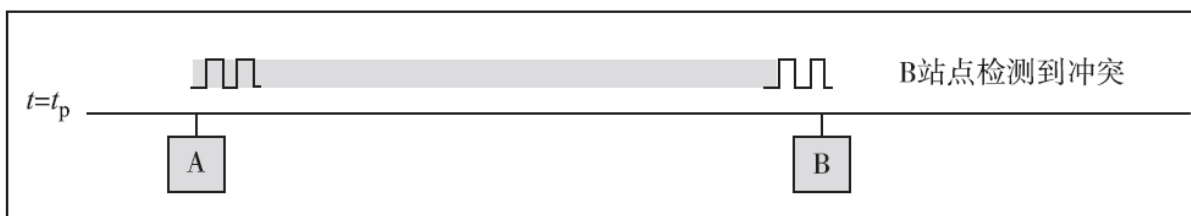


图 6-13 站点B检测到冲突

4) 再过了相当于一个往返传播时间 (也就是自站点A发送数据开始, 过了两个传播延时 $2t_p$) 后, 从站点B发来的阻塞信号帧也到达了站点A, A也知道发生冲突了, 停止数据发送, 如图6-14所示。此时站

点B会停止阻塞信号帧的发送，而站点A又会发送一个阻塞信号帧。最后，A和B都停止了数据发送，介质回归空闲状态。

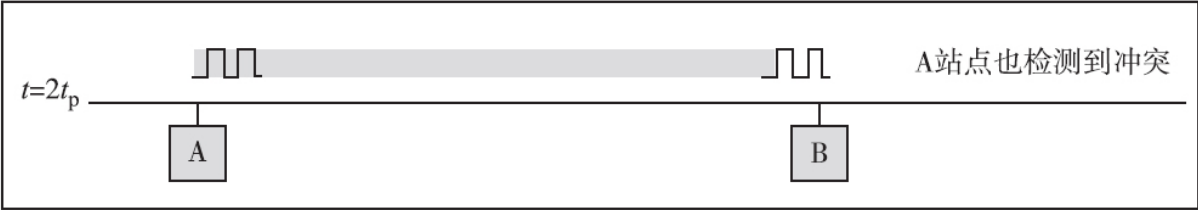


图 6-14 站点A也检测到冲突

6.3.3 冲突避让原理

CSMA/CD除了可以检测到冲突的发生外，还可以在冲突发生时进行随机延迟，以尽量避免再次发生冲突。这时会涉及一个避让延迟问题，也就是当检测到冲突时，到底要延迟多长时间再重传。这就是CSMA/CD的截断二进制指数退避算法（truncated binary exponential backoff）。在这种退避算法中，每次检测到冲突时避让（也就是名称中的“截断”含义，即停止发送数据）的时间是以时隙数为单位的随机时间，但不同次冲突时所能延迟的最大时隙数是不一样的，而且是呈指数关系递增的，所以才称之为二进制指数退避算法。因为当出现线路冲突时，如果冲突的各站点都采用同样的退避间隔时间，则很容易产生二次、三次的碰撞。

CSMA/CD的重传避让流程如图6-15所示。下面是具体的解释。

- 1) 首先，站点会初始化当前帧的重传次数N为0，并开始侦听介质状态。如果介质处于忙状态，则等待；在等到介质处于空闲状态后再等待一个帧间发送间隙的时间（IFG，为9.6μs），以使网络中所有接收站点有时间准备好接收下一个帧。

- 2) 然后，站点开始按次序传送帧，同时继续侦听介质状态。

3) 如果传到某个帧时侦听到发生了冲突，则该站点立即停止传输，开始冲突处理流程，发送一个用于加强冲突的阻塞信号，并把该帧的重传次数 N 递增1。

4) 到了这里，就要对 N 值进行判断了，因为不同的重传次数可以延迟的时隙数不一样。因为如果所有站点在发生冲突后都立即重传，则肯定又会发生另外一个冲突。所以一个正确的流程可确保一个同时重传的情况很少发生，在以太网中所采用的方案就是使用一个随机的避让时间，使每个站点选择一个随机数来乘以时隙长度（ $51.2\mu\text{s}$ ），在企图重传前必须等待这个随机的时长。

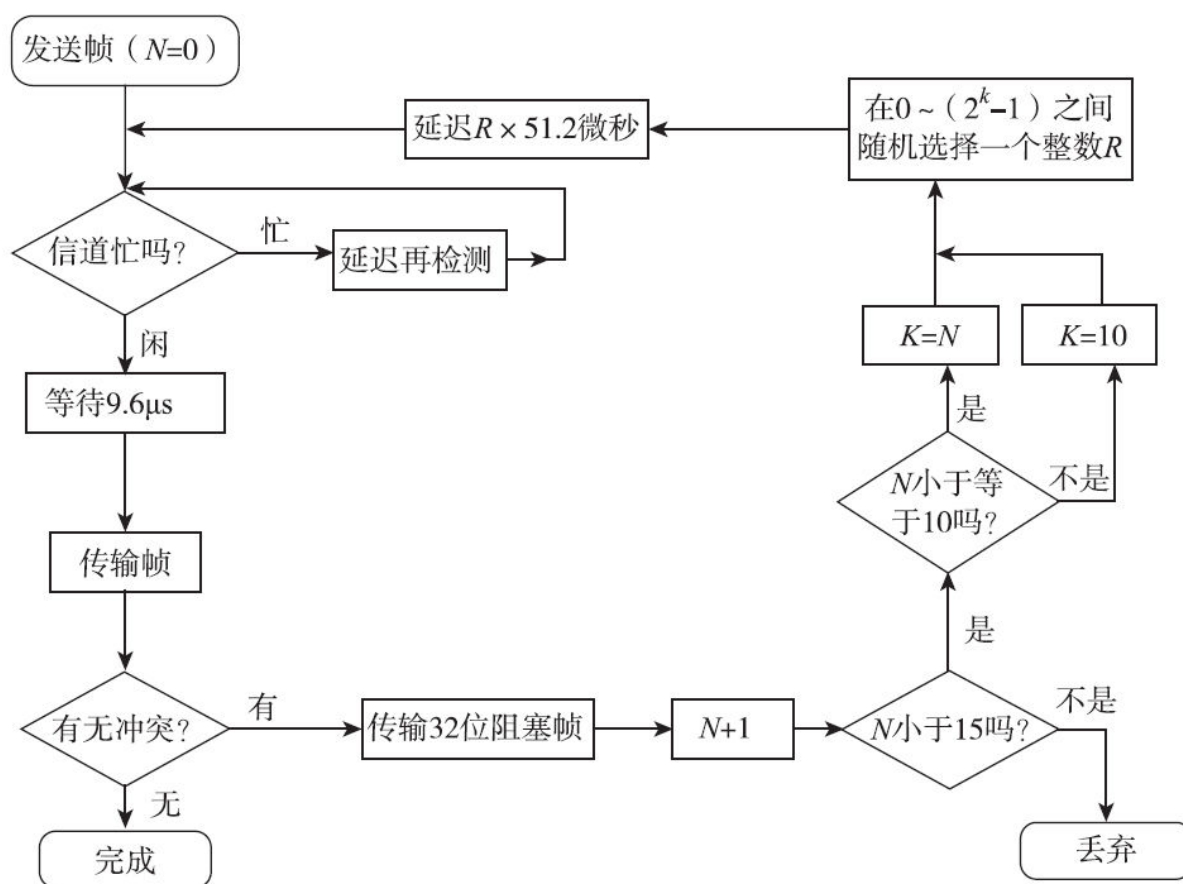


图 6-15 CSMA/CD避让原理示意图

发生冲突时，CSMA/CD可以计算出重传的次数（图中的N），并规定当N（重传次数）达到15时，该帧会被直接丢弃；当N小于15时，再判断N是否 ≤ 10 ，如果不是，则把K的值置为10，否则 $K=N$ ，然后根据公式 $R \times 51.2\mu s$ 来计算要延迟的时间。其中R是在 $0 \sim 2^K - 1$ 之间的一个随机整数，而 $K=N$ ，N为传输次数（ $K=N \leq 10$ ）。也就是，每个帧传输的次数是有限制的，那就是16，即 $N=15$ ，超过这个值，则会直接丢弃这个帧，放弃再次重传，并生成一个日志。通常在一个负载不是很重的网络中，是不会丢弃帧的，因为很难有重传15次后仍不能传输成功的。

每一次重传，站点都会构建一组可以选择的避让时隙数组 $\{0, 1, 2, 3, 4, 5, \dots, L\}$ ，其中 $L=2^K - 1$ ，且 $K \leq 10$ 。每次重传时所选择的避让时隙个数R就是从以上数组中选择的。例如，在发生两次冲突后， $N=2$ ，所以 $K=2$ ，得到的数组就是 $\{0, 1, 2, 3\}$ （ $2^2 - 1$ ），表示在发生两次冲突后，所需避让的时隙数有四个选择，也就是在 $0 \sim 3$ 个时隙中选择一个值，即 $\{0, 51.2\mu s, 102.4\mu s, 153.6\mu s\}$ 。

图6-16所示就是以上发生两次冲突时的避让示例。首先，站点A开始进行数据传输，然后站点B也开始数据传输；在站点A发送第二个帧的过程中与站点B发送来的数据发生冲突，此时站点A和站点B都会生成一个重传次数数组 $\{0, 1\}$ （因为此时只是发生了第1次冲突， $N=1$ ，

所以 $0 \sim 2^1 - 1 = 0 \sim 1$ ），并且可能都随机选择延迟一个时隙再发送数据。站点A和站点B在延迟一个时隙后再次同时发送数据时，再次发生冲突。此时站点A和站点B都会重新生成一个重传延迟时隙数组 $\{0, 1, 2, 3\}$ （因为此时只是发生了第2次冲突， $N=2$ ，所以 $0 \sim 2^2 - 1 = 0 \sim 3$ ）。但此时站点A选择延迟的时隙数为0，也就是不延迟，立即重传，而站点B选择是延迟2个时隙数后再重传数据。

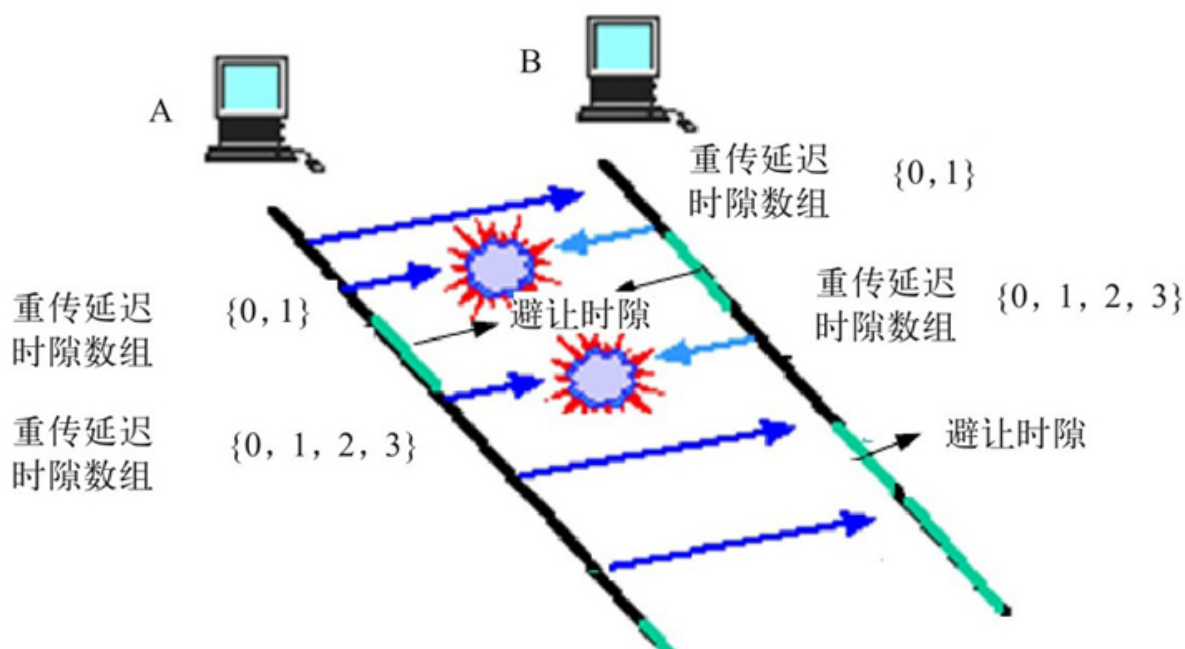


图 6-16 发生两次冲突时的避让示例

同理，在三次冲突后， $N=3$ ，可选择的避让时隙数组为 $\{0, 1, 2, 3, 4, 5, 6, 7\}$ ($2^3 - 1$)，也就是可以在等待以上其中8个之一可选时隙数长度后再重传数据。四次冲突之后， $N=4$ ，可选择的避让时隙数组为 $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\}$ (2^4

-1)，也就是可以在等待以上其中16个之一可选时隙数长度后再重传数据。

6.3.4 CSMA/CD的不足

使用CSMA/CD介质访问控制的一个缺点就是，当LAN中的互连的每台计算机都只有少量数据需要传输时，网络中每个站点对介质的共享都几乎是公平的；但是如果一个站点需要发送大量的数据时（如一个站点担当高质量视频源的情况下），这时就可能出现一个站点长时间控制整个LAN的情形。如图6-17所示，计算机A控制着计算机B。最初，两台计算机都有数据在传输。A先进行数据传输，然后A和B同时试图传输，发生了冲突。此时B选择了一个比A更大一些的等待时隙数来进行重传。A开始重传，这时有一个短时间的暂停，然后A和B都试图恢复传输。

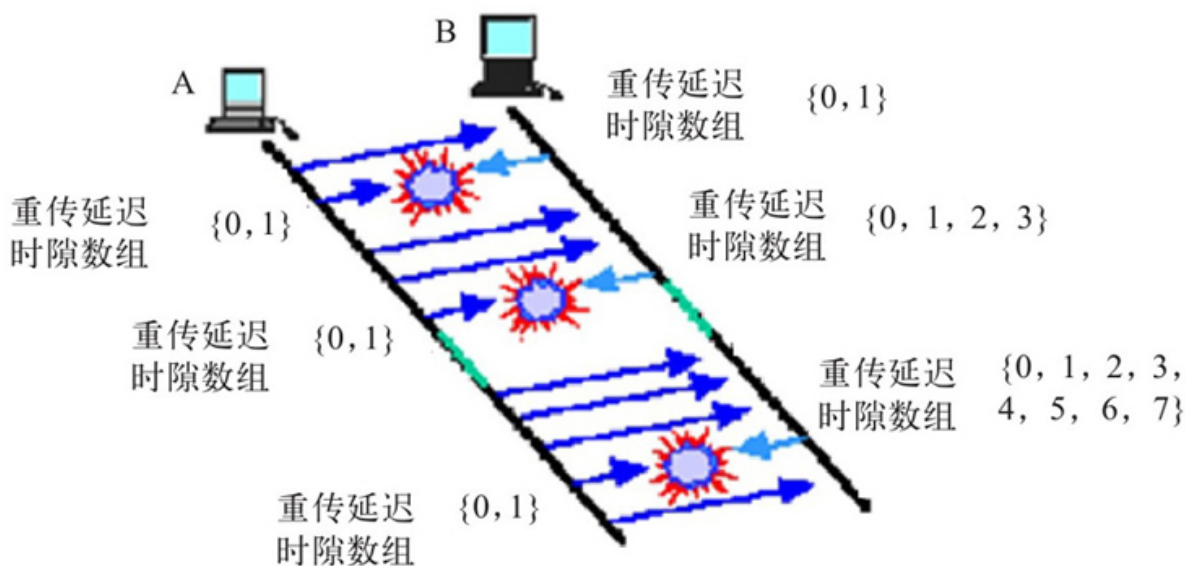


图 6-17 一个站点控制整个LAN共享介质的示例

6.4 局域网标准及以太网帧格式

在局域网中，绝大多数标准都是由IEEE（电气和电子工程师协会）802委员会制定的IEEE 802系列标准，其中在企业网络中应用最广的就是以太局域网标准IEEE 802.3。后来IEEE 802系列标准被ISO接受为正式的国际标准，标准名为ISO 8802。

6.4.1 IEEE 802系列局域网标准

1985年IEEE公布了应用于局域网的IEEE 802标准文本，同年为美国国家标准局（ANSI）采纳作为美国国家标准。后来，国际标准化组织（ISO）经过讨论，建议将802标准定为局域网国际标准。目前，IEEE 802主要包括的标准如表6-1所示，在许多标准中又有许多规范，就像IEEE 802.3标准中就包括了许多不同的以太网规范一样，这些在本章后面再具体介绍。这些802分委员会标准之间，以及各自与OSI参考模型之间的关系如图6-18所示。

表 6-1 主要 IEEE 802 标准

IEEE 802.1	局域网概述，局域网体系结构，网络管理和网络互连
IEEE 802.2	逻辑链路控制 LLC
IEEE 802.3	CSMA/CD 总线介质访问控制子层与物理层规范
IEEE 802.4	令牌总线（Token Bus）介质访问控制子层与物理层规范
IEEE 802.5	令牌环（Token Ring）介质访问控制子层与物理层规范
IEEE 802.6	城域网（MAN）介质访问控制子层与物理层规范
IEEE 802.7	宽带技术咨询和物理层课题与建议实施
IEEE 802.8	光纤技术咨询和物理层课题
IEEE 802.9	综合语音 / 数据服务的访问控制方法和物理层规范
IEEE 802.10	局域网安全性规范
IEEE 802.11	无线局域网访问控制方法和物理层规范
IEEE 802.12	100VG-AnyLAN 星型快速局域网访问控制方法和物理层规范
IEEE 802.14	协调混合光纤同轴（HFC）网络的前端和用户站点间数据通信的协议
IEEE 802.15	无线个人网技术标准，其代表技术是蓝牙（Bluetooth）

OSI参考模型

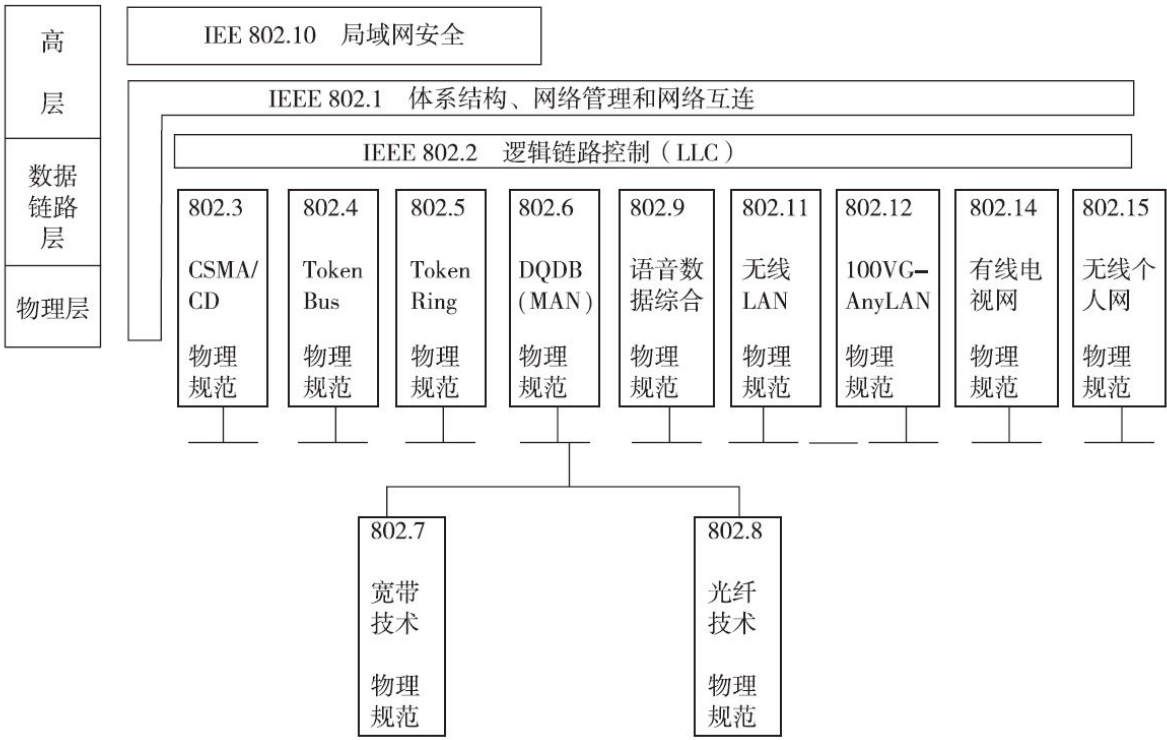


图 6-18 IEEE各标准之间，以及与OSI参考模型之间的关系

IEEE 802.1、802.2、802.3、802.11等这些主要局域网标准在本章后面有专门介绍，其他局域网标准的具体功能大家可以参考其他资料。

6.4.2 以太网帧格式综述

以太网（Ethernet）这个术语一般是指数字设备公司（Digital Equipment）、英特尔公司（Intel）和施乐公司（Xerox）在1982年联合公布的一个标准（实际上它是第二版本，即Ethernet II，第一版本早在1972年就在施乐公司帕洛阿尔托研究中心PARC里产生了）。

Ethernet II采用CSMA/CD介质访问控制方法，传输速率为仅10 Mbps。1985年，IEEE（电子电气工程师协会）的802委员会公布了一个系列的以太网标准，其中802.3专门针对以太网、802.4针对令牌总线网络、802.5针对令牌环网络，但这三种网络的共同特性由802.2标准来定义，那就是802网络共有的逻辑链路控制（LLC）。不幸的是，802.2和802.3定义了一个与Ethernet II以太网不同的帧格式，加上1983年Novell为其Netware开发的私有帧，这些给以太网造成了一定的混乱，也给我们学习以太网带来了一定的影响。

在以太网帧格式方面上，主要有过以下的六种。在此先仅对这些以太网帧格式进行简单介绍，具体的字段说明将在后面介绍。

（1）Ethernet I

这是最原始的一种以太网MAC帧格式，是由Xerox公司提出的3Mbps CSMA/CD以太网标准的封装格式，后来在1980年由DEC、Intel

和Xerox三家公司发布。其中最关键的一个字段就是类型（Type）字段，以便它可以支持多种网络层协议包，如TCP/IP协议、IPX/SPX、Apple Talk等。但该版本当时应用并不广泛，随后基本被新的Ethernet II版本取代。

(2) Ethernet II

这是由DEC、Intel和Xerox三家公司在1982年发布的，即DIX2.0的以太网帧格式。它主要更改了Ethernet I的电气特性和物理接口，在帧格式上并无变化，如图6-19所示。其中除数据字段（46~1500个字节）外，其他均为MAC子层的协议头和协议尾（此时并没有划分LLC子层，所以没有LLC子层头部分），最小帧（包括帧头和帧尾）长度为64字节，最长为1518字节。Ethernet II出现后迅速取代Ethernet I成为以太网事实标准。

6	6	2	46~1500	4 字节
目的 MAC 地址	源 MAC 地址	类型	数据	FCS

图 6-19 Ethernet II帧格式

(3) Ethernet 802.3 raw

这是Novell公司在1983年公布的专用以太网标准帧格式（仅支持IPX/SPX这一种协议）。该格式以当时尚未正式发布（所以加了一个raw，就是“原始”的意思）的IEEE 802.3标准为基础，具体的帧格式如

图6-20所示。相对Ethernet II帧来说，就是多了一个2字节的0xFFFF，用于标识这个帧是Novell Ethernet类型的。由于总的以太网帧最小和最大长度不变（仍分别为64字节和1518字节），所以数据字段中的最小和最大长度也相应减小2字节，为44~1498字节。但是当两年后IEEE正式发布802.3标准时，情况发生了变化，因为IEEE在802.3帧头中又加入了802.2 LLC头，这使得Novell的RAW 802.3格式与正式版的IEEE 802.3标准互不兼容。

6	6	2	2	44~1498	4 字节
目的 MAC 地址	源 MAC 地址	长度	0xFFFF	数据	FCS

图 6-20 Ethernet 802.3 raw帧格式

(4) Ethernet 802.3 SAP

这是IEEE在1985年公布的Ethernet 802.3的SAP版本以太网帧格式，是IEEE发布的第一个以太网帧格式版本，如图6-21所示。SAP是服务访问点的意思，表示添加了LLC帧头部，包含目的服务访问点

（DSAP）、源服务访问点（SSAP）和控制（Control）这三个字段。另外，它将Ethernet II帧头的Type（类型）字段替换为帧Length（长度）字段。因为新添加了DSAP、SSAP和控制这三个各占1字节的字段，所以数据字段的长度范围也要相应调整为43~1497字节，这也是类型字段的取值范围。这个以太网帧格式版本可以说是一个过渡版本，

因为IEEE在后面很快又进行了更新，相继发布了802.3/802.2 LLC和802.3/802.2 SNAP版本。

6	6	2	1	1	1	43 ~ 1497	4 字节
目的 MAC 地址	源 MAC 地址	长度	DASP	SSAP	控制	数据	FCS

图 6-21 Ethernet 802.3 SAP帧格式

(5) 802.3/802.2 LLC

这是IEEE 1997年正式发布的802.3标准，由Ethernet 802.3 SAP版本发展而来，且帧格式也一样。但这是IEEE正式划分LLC子层后的第一个以太网标准，第一次把原来DSAP、SSAP和控制这三个字段当成LLC头。

(6) 802.3/802.2 SNAP

这是IEEE为保证在802.3/802.2 LLC标准上支持更多的上层协议同时，更好地支持IP协议而于1998年发布的扩展以太网帧标准。SNAP（SubNetwork Access Protocol，子网访问协议是一种可以传输多种协议包的网络访问协议。

与802.3/802.2 LLC帧格式一样，802.3/802.2 SNAP也带有LLC头（如图6-22所示），但是扩展了LLC属性：一是新添加了一个2字节的类型字段（注意，此类型字段与Ethernet II版本中帧格式的类型字段是不一样的），用于标识更多的上层协议类型；另外新添加了一个3字节

的OUI（组织唯一标识，通常是全为0）字段，用于代表发布所选上层协议的组织。同样因为总的以太网帧最小和最大帧长度是不变的（仍分别为64字节和1518字节），除了数据字段外，其他字段的总长为26字节。所以数据字段的长度范围为38~1492个字节，这也是长度字段的取值范围。

6	6	2	1	1	1	3	2	38~1492	4 字节
目的 MAC 地址	源 MAC 地址	长度	DASP	SSAP	控制	OUI ID	类型	数据	FCS

图 6-22 802.3/802.2 SNAP帧格式

我们在第3章就介绍到了，从应用层发送来的PDU（协议数据单元），每经过一层都要把上层发送来的整个PDU作为下层PDU的Data（数据）字段部分，然后在其前面加上对应层的协议头（Header）和协议尾（Tail，并不是一定要加）。

在以上六种以太网帧格式中，Ethernet I、Ethernet II和Ethernet 802.3 raw这三种以太网帧仅包含MAC帧部分，而在正式的IEEE 802.3 SAP以太网标准发布后，因为同时发布了LLC子层的IEEE 802.2标准，这时的以太网帧格式其实就包括了MAC帧和LLC帧两部分了。来自网络层的数据包到达数据链路层后要经过LLC和MAC这两个子层的两次先后封装。MAC帧是把整个LLC帧当成Data字段，然后再加上自己的MAC帧头和帧尾，就构成了最终的MAC帧，也就是我们通常所说的以太网帧，整个以太网帧的封装流程如图6-23所示。

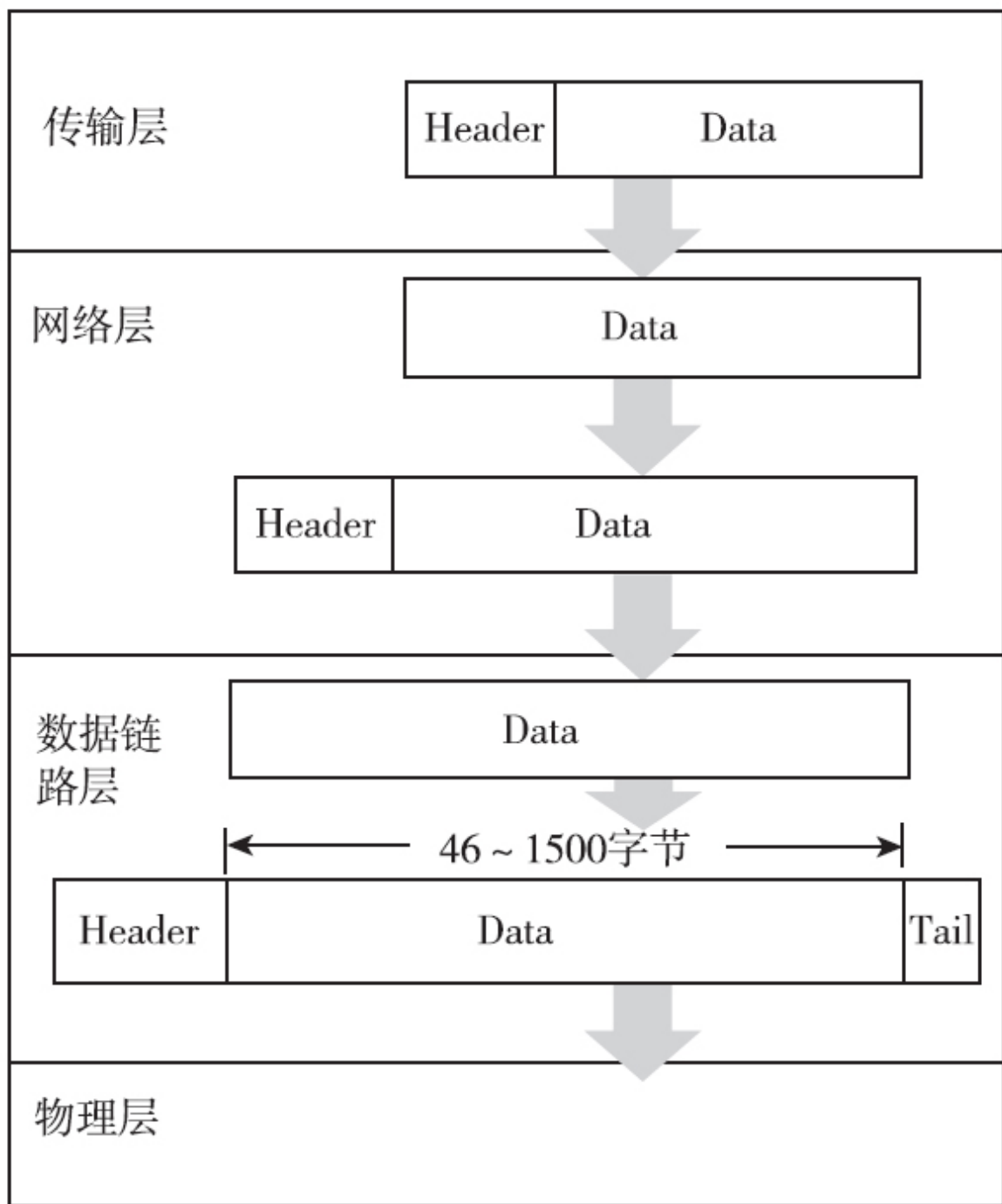


图 6-23 PDU的各级封装方式示意图

在以上这六种以太网帧格式中，曾经或者现在主流应用的是 Ethernet II、802.3/802.2 LLC和802.3/802.2 SNAP这三种，它们的格式对

比如图6-24所示，各具体字段将在后面介绍。

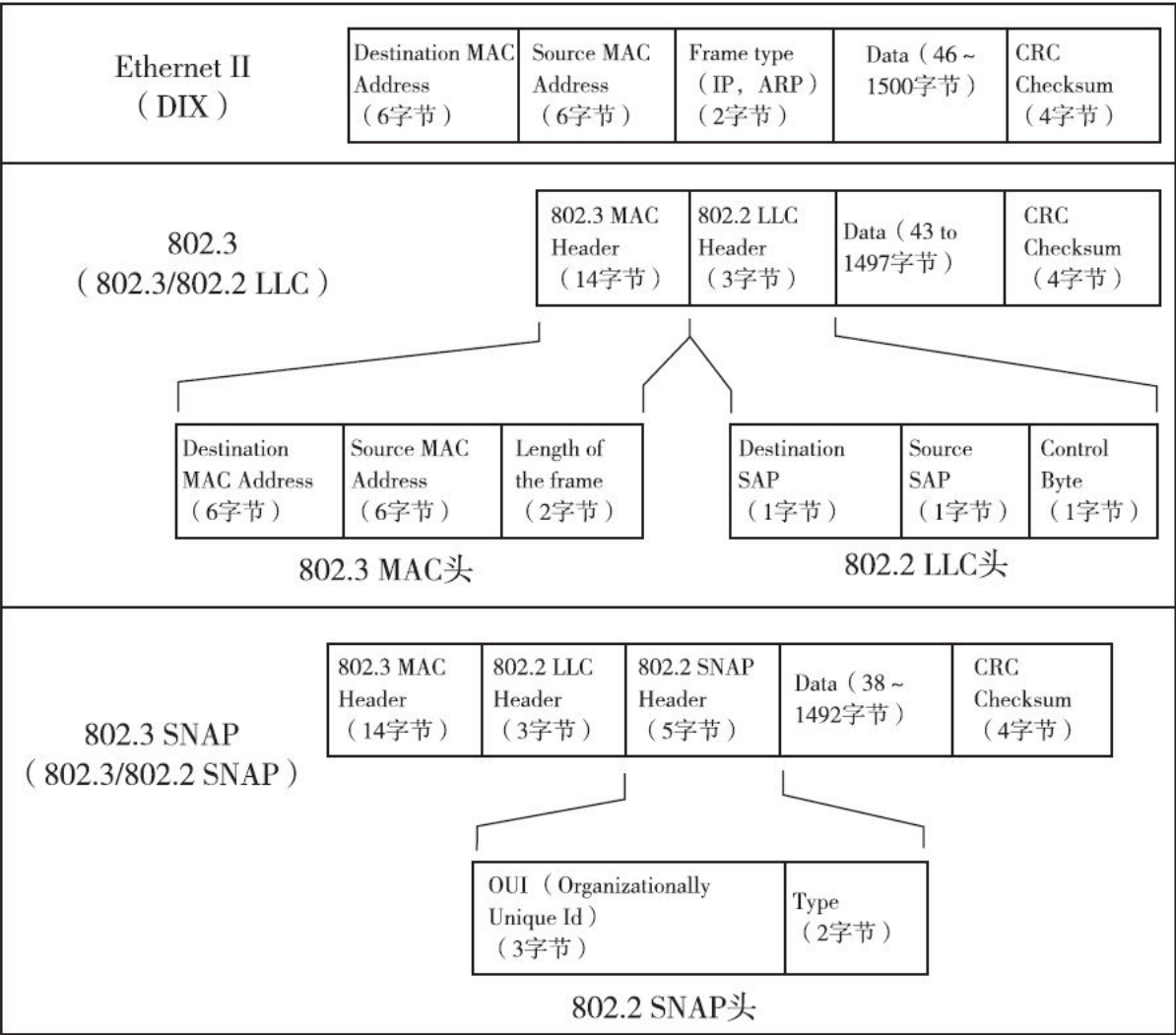


图 6-24 三种主要以太网帧格式比较

6.4.3 以太网LLC帧头部格式

目前说的以太网帧，通常都会认为其帧头和帧尾包括LLC帧头、MAC帧头和MAC帧尾这三部分。上节就已说到，自Ethernet 802.3 SAP以太网标准发布后，在以太网帧中添加了802.2 LLC帧头，这一点也可以从图6-24中看出。在802.3系列以太网中，LLC帧头格式是一样的（共3字节长度），如图6-25示。本节要具体解释这三个字段的含义。

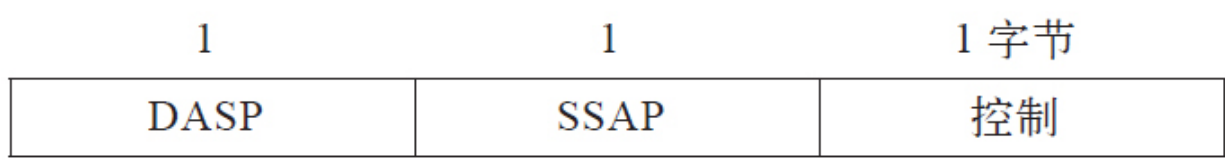


图 6-25 LLC帧头部格式

□DSAP：目的服务访问点（Destination Service Access Point）字段，指示数据接收方的LLC子层的SAP，占1字节（8位）。在以太网中，该字段的值固定为十六进制的0xAA。

□SSAP：源服务访问点（Source Service Access Point）字段，指示数据发送方的LLC子层的SAP，占1字节（8位）。在以太网中，该字段的值也固定为十六进制的0xAA。

□控制：Control字段，用于指示数据链路层所用的服务类型，占1字节（8位）。在以太网中都是采用无连接服务，所以固定值为十六进

制的0x03。

6.4.4 以太网SNAP头部格式

从图6-24可以看出，在802.3/802.2 SNAP以太网帧中，除了添加了3字节的LLC头部外，还添加了5字节的SNAP头部作为LLC扩展部分，如图6-26所示。SNAP头部包括OUI ID和类型这两个字段，本节将具体解释这两个字段的含义。

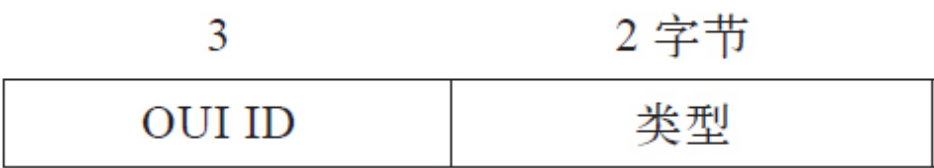


图 6-26 SNAP头部格式

□OUI: 组织唯一标识（Organizationally UniqueIdentifier）字段，指示帧中Data字段数据报所对应协议的发布公司的标识，占3字节（24位）。如果是IEEE公司发布的以太网协议类型，则此字段固定为0x000000。

□Type: 类型字段，其实是协议ID字段，指示帧中数据字段数据报所对应协议类型，占2字节（16位）。不仅支持在以太网MAC帧中Type（类型）字段中所支持的802.3以太网协议，还支持802.4、802.5、802.11和其他802系列协议，甚至支持像FDDI（Fiber Distributed Data Interface，光纤分布式接口）这样的非802协议。如果是IEEE公司发布

的以太网协议类型，则此字段的值根据表6-2来选择。有关这些协议的具体说明可参见其他相关资料。

表 6-2 常见以太网类型协议

协议 ID	以太网协议
0x0800	Internet Protocol, Version 4 (IPv4)
0x0806	Address Resolution Protocol (ARP)
0x0842	Wake-on-LAN Magic Packet
0x1337	SYN-3 heartbeat protocol (SYNdog)
0x22F3	IETF TRILL Protocol
0x6003	DECnet Phase IV
0x8035	Reverse Address Resolution Protocol (RARP)
0x809B	AppleTalk (Ethertalk)
0x80F3	AppleTalk Address Resolution Protocol (AARP)
0x8100	VLAN-tagged frame (IEEE 802.1Q)
0x8137	Novell IPX (alt)
0x8138	Novell
0x8204	QNX Qnet
0x86DD	Internet Protocol, Version 6 (IPv6)
0x8808	MAC Control
0x8809	Slow Protocols (IEEE 802.3)
0x8819	CobraNet
0x8847	MPLS unicast
0x8848	MPLS multicast
0x8863	PPPoE Discovery Stage
0x8864	PPPoE Session Stage
0x886F	Microsoft NLB heartbeat ^[3]
0x8870	Jumbo Frames
0x887B	HomePlug 1.0 MME
0x888E	EAP over LAN (IEEE 802.1X)
0x8892	PROFINET Protocol
0x889A	HyperSCSI (SCSI over Ethernet)
0x88A2	ATA over Ethernet
0x88A4	EtherCAT Protocol
0x88A8	Provider Bridging (IEEE 802.1ad)
0x88AB	Ethernet Powerlink
0x88CC	LLDP
0x88CD	sercos III
0x88D8	Circuit Emulation Services over Ethernet (MEF-8)
0x88E1	HomePlug AV MME
0x88E3	Media Redundancy Protocol (IEC62439-2)

(续)

协议 ID	以太网协议
0x88E5	MAC security (IEEE 802.1AE)
0x88F7	Precision Time Protocol (IEEE 1588)
0x8902	IEEE 802.1ag Connectivity Fault Management (CFM) Protocol / ITU-T Recommendation Y.1731 (OAM)
0x8906	Fibre Channel over Ethernet
0x8914	FCoE Initialization Protocol
0x9000	Configuration Test Protocol (Loop) ^[4]
0x9100	Q-in-Q

6.4.5 以太网MAC帧

MAC帧通常被认为是数据链路层帧，是在MAC子层实体间交换的协议数据单元（PDU）。根据图6-24所示，可以把所有以太网MAC帧的格式统一表示为图6-27所示，前面两节介绍的LLC帧头和SNAP头部与上层来的数据报一起均封装在数据字段中。

7	1	6	6	2	38~1500	4字节
前导	帧起始	目的MAC地址	源MAC地址	长度/类型	数据	FCS

图 6-27 MAC帧格式

（1）前导

前导（Preamble）字段占7个字节，由1和0交互构成（如10101010.....），用于使PLS子层电路与收到的帧达到时钟同步。

（2）帧起始

帧起始（Start-of-Frame Delimiter，SFD）字段占1个字节，前6位也是1和0交互构成，最后两位是连续的1，即10101011，表示一个帧的开始。前导码的作用是使接收端能根据1、0交互的比特模式迅速实现比特同步，当检测到连续两位1（即读到帧起始定界符字段SFD最末两位）时，便将后续的信息递交给MAC子层。

说明 在以上两个字段中，在早期的Intel和Xerox公司开发的以太网标准中是把SFD字段并入到了Pre字段中，所以那时的MAC帧格式中没有SFD字段。只有后面由IEEE发布的以太网标准中才出现了SFD字段。但Pre和SFD这两个字段只是用来提醒接收端新的一帧到来了，并不计入MAC帧大小中。

(3) 目的MAC地址/源MAC地址

目的MAC地址（Destination Address，DA）和源MAC地址（Source Addresses，SA）字段各占6个字节，分别用于标识接收站点的MAC地址和发送站点的MAC地址。它们可以是单播MAC地址，也可以是组播地址或广播MAC地址。地址字段最高位为0表示单播MAC地址，该地址仅指定网络上某个特定站点；地址字段最高位为1、其余位不为全1表示组播MAC地址，该地址指定网络上给定的多个站点；地址字段为全1，则表示广播MAC地址，该地址指定网络上所有的站点。

(4) 长度/类型

长度/类型（Length/Type）字段是一个二选一字段，也就是对于具体的以太帧来说，它的含义不一样，占2字节。在Ethernet I和Ethernet II以太网帧中，该字段为类型（Type）字段，指出帧中数据字段中的数据类型，这些类型的值如表6-2所示，总大于1536（对应的十六进制为0x600）；如果是IEEE 802.3（包括Ethernet 802.3 raw、Ethernet 802.3

SAP、802.3/802.2 LLC和802.3/802.2 SNAP)以太网帧,则该字段为长度(Length)字段,值总小于或等于1500(对应的十六进制为0x5DC)。对照图6-24可知,在IEEE 802.3以太网帧中,数据字段的长度为38~1500字节。

说明 上面的DA、SA和Length/Type这三个字段组成MAC帧头部。Pre和SFD这两个字段通常不认为是MAC帧头部的组成部分。

(5) Data

数据(Data)字段对于不同的以太网帧所包括的内容也不一样,对于Ethernet I、Ethernet II和Ethernet 802.3 raw以太网帧,它就是从网络层来的数据包;而对于Ethernet 802.3 SAP、802.3/802.2 LLC和802.3/802.2 SNAP以太网帧,则是LLC帧全部内容,包括LLC帧头和来自网络层的数据包。也正如此,“数据字段”长度范围也各不一样,具体如下(可参见图6-19~图6-22中的数据字段字节范围标注):

- ☐ Ethernet I、Ethernet II帧数据字段长度范围为46~1500字节;
- ☐ Ethernet 802.3 raw帧数据字段长度范围为44~1498字节;
- ☐ Ethernet 802.3 SAP和802.3/802.2 LLC帧数据字段长度范围为43~1497字节;
- ☐ Ethernet 802.2 SNAP帧数据字段长度范围为38~1492字节。

总体而言，该字段长度范围为38~1500个字节。但无论如何，总的MAC帧长度最小为64字节，最长为1518字节（不包括“前导”字段和“帧起始字段”），如果不够64字节时，要在“数据”字段中加上PAD填充字段。

注意 这里所说的38~1500字节长度是在没有经过IEEE 802.1Q VLAN协议封装时的长度范围，如果封装了VLAN协议，则因为VLAN标识占用了4字节，所以就整个以太网帧来说，数据字段的取值范围就为34~1500字节。有关IEEE 802.1Q VLAN协议将在本章后面具体介绍。

（6）Frame Check Sequence（FCS）

帧校验序列（FCS）字段占4个字节，包括32位的循环冗余校验（CRC）值，由发送端对MAC帧自DA字段到Data字段间（不包括Pre和SFD这两个字段）的二进制序列生成校验和，然后通过接收端对所接收的帧以上部分（自DA字段到Data字段间，不包括Pre和SFD两个字段）进行重新计算，看两次校验的结果是否一样即可以得出所检验的帧在传输过程中是否已被破坏。有关CRC校验原理参见第4章。

6.5 标准以太网规范及体系结构

自本节开始，就要全面、系统、深入地介绍我们天天打交道的以太网局域网了。**10Mbps**以太网是第一个真正走入实质应用的以太网标准，尽管现在早已不再使用，但是后面的所有以太网标准都是基于这个标准改进、开发的，所以我们还是很有必要全面地了解这个早期的以太网标准。

6.5.1 标准以太网规范

标准以太网（也就是十兆以太网），以及后面将要介绍的快速以太网（百兆以太网）、千兆以太网和万兆以太网规范都是在**IEEE 802.3**家族标准中发布的。本节先介绍标准以太网。

标准以太网最初使用的是同轴电缆作为传输介质（此时为总线型拓扑结构），后来为了节省成本，又开发了基于双绞线标准以太网规范（此时使用集线器作为集中连接设备），再后来又为了提高传输距离，又诞生了光纤标准以太网规范。

具体来说，标准以太网规范中主要包括：使用**50Ω**粗同轴电缆连接的**10Base5**（对应**IEEE 802.3**标准），使用**50Ω**细同轴电缆的**10Base2**（对应**IEEE 802.3A**标准），使用**75Ω**同轴电缆的**10Broad36**（对应

IEEE 802.3B标准），使用双绞线电缆的10Base-T（对应IEEE 802.3I标准）和使用光纤的10Base-F（对应IEEE 802.3J标准）。

这些规范名称都有其特殊含义的：对于采用同轴电缆作为传输介质的10Mbps以太网规范，该委员会给出了他们所采用的规范名称格式为：

<数据传输速率（Mbps）> <信号传输模式> <最大段长度（百米）>

如10Base-5、10Base-2、10Broad36中前面的10表示数据传输速率为10Mbps，Base表示信号采用基带传输方式，最后面的5、2、36则分别表示这几种标准中单段介质的最大长度为500m、200m（实际上是185m）和3.6km。但10Base-T有些例外，其中的T表示双绞线（Twisted Pairwire），10Base-F中的F表示传输介质为光纤（Fiber）。

下面简单介绍这几种标准以太网规范。

（1）10Base-5和10Base-2

10Base-5（或者10Base5）和10Base-2（或者10Base2）这两个规范都是在使用阻抗为50Ω同轴电缆作为传输介质的总线型拓扑结构网络中，均采用基带传输模式的曼彻斯特编码（有关这种编码的特点参见第4章相关内容）。两者的主要区别在于10Base-5使用粗同轴电缆（单

段电缆的最大长度为500m)，10Base-2使用细同轴电缆（单段电缆的最大长度为180m）。

（2）10Broad36

10Broad36是IEEE 802.3中唯一针对宽带系统（如有线广播、有线电视网络）的规范，在采用阻抗为75Ω同轴电缆的总线型网络拓扑结构网络中使用。它也采用基带传输模式的曼彻斯特编码，最大的端到端距离为3.6km。

（3）10Base-T

10Base-T（或者写作10BaseT）规范应用于使用双绞线作为传输介质的星型拓扑结构以太网络中，是当时以太局域网中应用最广泛的以太网规范。它也采用基带传输模式的曼彻斯特编码，单段电缆的最大长度为100m。

（4）10Base-F

10Base-F（或者写作10BaseF）规范使用光纤作为传输介质，包括10Base-FL、10Base-FB和10Base-FP三种子规范，均采用基带传输模式的曼彻斯特编码，但它们的设计用途各不相同。

□10Base-FL（或者写作10BaseFL）子规范设计用于与当时的FOIRL（Fiber Optic Inter Repeater Link，光纤中继器间连接）协议协同

工作，其目的就是用来替换FOIRL。在与FOIRL协同工作时单段光纤长度最大为1000m，纯10Base-FL单段光纤长度最大为2000m。显然，10Base-FL是用于光纤中继器间的连接。

□10Base-FB（或者10BaseFB）10Mbps光纤以太网子规范设计用于光纤骨干网（Fiber-Backbone）中同步信令的传输，用于将其他网段通过中继器连接到本地网络（不是用于用户连接的，用户光纤连接采用的是10Base-FP子规范），单段光纤长度最大为2000m。显然，10Base-FB是用于集线器，或交换机与光纤中继器间的连接。

□10Base-FP（或者写作10BaseFP）是一种用于连接无源光纤（Fiber-Passive）器件（这里的“无源”是指不需要接外部电源就可直接工作）的以太网子规范，可以无需中继器即可连接大量用户（通过集线器或交换机）组成星形拓扑结构网络，单段光纤长度最大为500m。显然，10Base-FP是用于用户与集线器，或交换机间的连接。

6.5.2 标准以太网物理层结构

因为在IEEE 802.3以太网中，MAC子层是直接与物理层直接连接的，关系非常密切，所以把以太网中的物理层结构放在本章介绍。

标准以太网接口的物理层和数据链路层结构（注意，这里是基于逻辑功能上的划分，不是基于物理结构上的划分）如图6-28所示。从图中可以看出，标准以太网的物理层包含了以下三个部分：MAU、AUI和PLS。下面分别予以介绍。

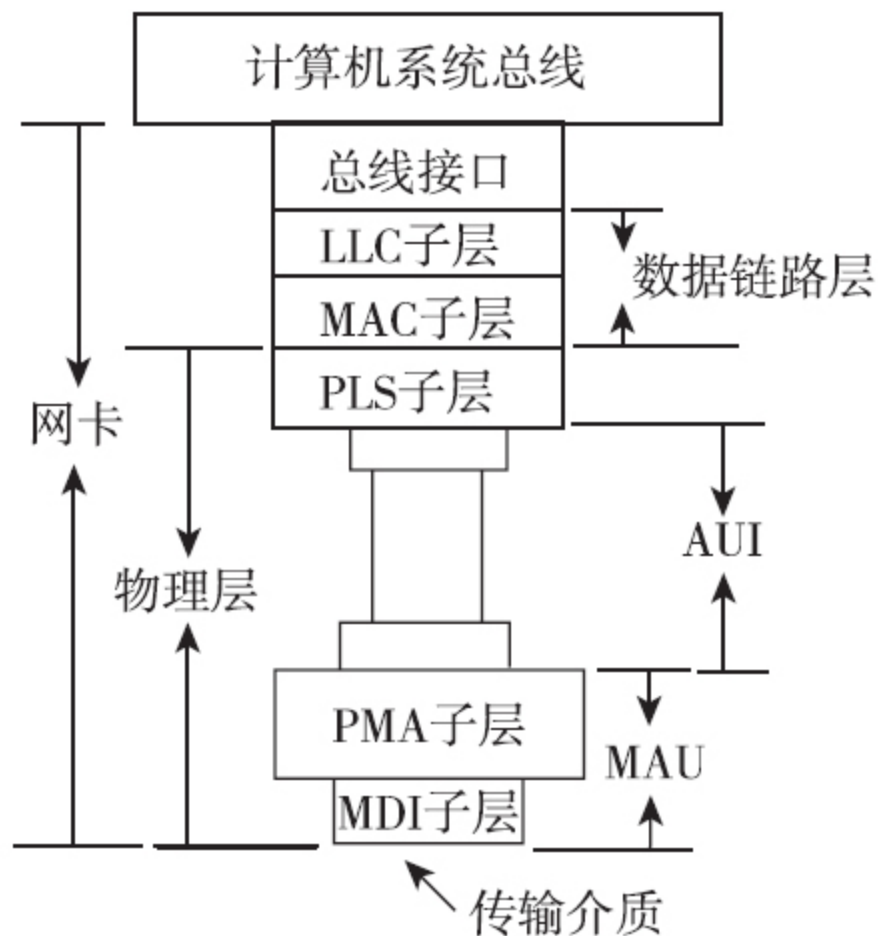


图 6-28 10Mbps以太网物理层和数据链路层结构

1.介质连接单元

MAU (Medium Attachment Unit, 介质连接单元) 是网络接口用来直接与传输介质 (如同轴电缆、双绞线, 光纤等) 连接的那部分结构, 发挥作用的是里面的收发器 (transceiver) 芯片。在该部分中又包括了PMA (Physical Medium Attachment, 物理介质连接) 和介质相关接口 (Medium Dependant Interface, MDI) 两个子层的功能, 在网络接口和传输介质之间提供机械连接和电气特性接口。

MDI子层是标准以太网接口物理层中的最低层，直接负责处理网络接口与传输介质的连接，其实就是网络接口连接器。它定义了电缆、连接电缆的连接器，以及电缆两端的终端负载的特性。因为标准以太网可以有多种不同传输介质类型，所以也就对应有多种MDI连接器，这也就是MDI中“相关”的含义。如粗同轴电缆的MDI称为插入式分接头（其实又称AUI接口，如图6-29所示），细同轴电缆的MDI是BNC连接器（如图6-30所示），双绞线以太网的MDI称为RJ—45连接器（如图6-31所示），光纤以太网的MDI可以是ST或者SC连接器等（分别如图6-32、图6-33所示）。



图 6-29 粗同轴电缆AUI连接器



图 6-30 细同轴电缆BNC连接器



图 6-31 双绞线RJ-45连接器



图 6-32 ST光纤连接器

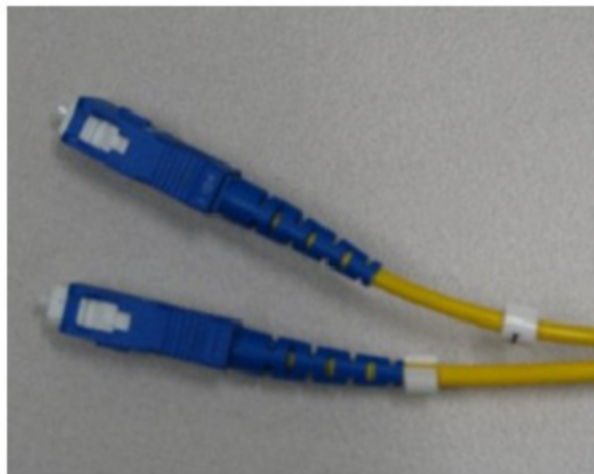


图 6-33 SC光纤连接器

PMA子层向PLS子层提供服务，负责处理与物理介质连接方面的功能（在收发器中实现），如串/并行传输的转换、检测冲突、超时控制（jabber control），以及收发比特流。其中的超时控制是指当检测到某个站点发送的数据超过设定的最长传输时间（在芯片中有专门负

责超时检测的引脚) 时, 即认为该站点出了故障, 接着就会自动禁止该站向总线发送数据。

2.连接单元接口

AUI (Attachment Unit Interface, 连接单元接口) 相当于网络接口收发器上的电缆。它定义了将MAU与PLS子层相连的电缆的机械和电气特性, 同时还定义了通过这个电缆所交换的信号的特性。AUI上的信号有4种: 发送和接收的曼彻斯特编码、冲突信号和电源信号。

注意 如果PLS子层与PMA子层处于同一个DTE设备中, 就不需要AUI和MAU了, 但DTE与同轴电缆的MDI子层还是需要的。

3.物理层信号子层

PLS (Physical Layer Signaling, 物理层信号) 子层为MAC子层服务, 是在网卡中实现的 (PMA子层和MDI子层是在收发器中实现的)。如果PLS子层与PMA子层不处在同一个设备中, 则它通过AUI与MAU连接。从其名称可以看出, PLS子层的主要功能是对物理层信号的处理, 具体包括以下两个方面:

□编码/解码: 发送时将MAC子层来的串行数据编码为曼彻斯特编码, 并通过收发器电缆发送给收发器; 接收时, 接收AUI发送来的曼彻斯特编码信号, 并进行解码, 然后以串行方式发送给MAC子层。

□载波侦听：确定介质是否空闲，然后把侦听到的载波侦听信号发送给MAC子层。

6.6 快速以太网规范及体系结构

100Mbps的快速以太网（Fast Ethernet）技术是由10Mbps标准以太网发展而来的，主要解决网络带宽在局域网络应用中的瓶颈问题。其协议标准为1995年颁布的IEEE 802.3u（100Base），可支持100Mbps的数据传输速率。

IEEE 802.3u在MAC子层仍采用了在IEEE 802.3标准以太网中的CSMA/CD作为介质访问控制协议，并保留了标准以太网的MAC和LLC帧格式。但是，为了实现100 Mbps的传输速率，在物理层做了一些重要的改进。例如，在编码上采用了效率更高的编码方式——4B/5B编码（在10Mbps以太网中采用的是曼彻斯特编码方式，但其编码效率比较低，有效信号只占1/2码字）。

6.6.1 快速以太网规范

IEEE 802.3u快速以太网标准中定义了以下三种不同的快速以太网规范：

□100Base-TX：采用两对（4根）芯线的5类（包括超5类）非屏蔽双绞线或者屏蔽双绞线作为传输介质

□100Base-T4: 采用四对（全部的8根）芯线普通3、4、5类（包括超5类）双绞线作为传输介质

□100Base-FX: 采用光纤作为传输介质

IEEE 802.3u标识符包括三块信息：其一，100表示传输速度100Mbps；其二，Base表示信号仍采用基带传输方式；其三，是关于网络段类型的阐述，T4表示要用四对双绞线，TX表示要用两对双绞线，FX表示要用两条光纤。下面是这三种快速以太网标准的介绍。

1.100Base-TX

100Base-TX快速以太网规范是采用两对芯线的双绞线电缆（可以是5类或超5类屏蔽或非屏蔽双绞线，但目前普遍采用的是成本更低，更易用布线的非屏蔽双绞线）的，其中一对用于发送数据，另一对用于接收数据。该标准直接用于取代标准以太网中的10Base-T和10Base-2规范。

100Base-TX规范中的MDI（介质相关接口）连接器有两种：对于非屏蔽双绞线（UTP），MDI子层连接器是8个引脚的RJ—45连接器；对屏蔽双绞线（STP），MDI子层连接器必须是IBM的STP连接器。

如果使用的是非屏蔽双绞线，100Base-TX规范中只使用了四对芯线中的两对，另外两对芯线没有使用。根据EIA/TIA-568B布线标准，

直通双绞网线（两端同时用TIA/EIA-568A，或者TIA/EIA-568B标准，用于不同设备的连接，如PC机到集线器、交换机的连接，交换机到路由器的连接等）RJ—45连接器的8个引脚与双绞线8条芯线的连接如表6-3所示。

表 6-3 100Base-TX 直通 UTP 网线水晶头引脚、信号功能与双绞芯线的连接分配表

引脚号	信号功能	芯线颜色
1	发送 +	橙白色
2	发送 -	橙色
3	接收 +	绿白色
4	保留	—
5	保留	—
6	接收 -	绿色
7	保留	—
8	保留	—

100Base-TX标准也支持特征阻抗为150Ω的5类屏蔽双绞线。屏蔽双绞线电缆使用D型连接器。直通STP网线DB-9连接器上引脚与双绞芯线的连接如表6-4所示。

表 6-4 100Base-TX 的 STP MDI 连接器引脚与双绞芯线的连接分配表

引脚号	信号功能	芯线颜色
10	公共地	电缆外壳
1	接收 +	橙色
2	保留	—
3	保留	—
4	保留	—
5	发送 +	红色
6	接收 -	黑色
7	保留	—
8	保留	—
9	发送 -	绿色

当要使用交叉电缆时（如同类设备普通端口上的级联），按照表6-5所示的跳线规则进行交叉网线（一端用TIA/EIA-568A标准，另一端用TIA/EIA-568B标准，用于同种设备的连接，如PC机到PC机的连接，交换机普通端口间的级联，PC机到路由器的连接，路由器与路由器间的连接等）制作。但是目前的以太网端口基本上都支持电缆跳线自动翻转功能，任何情况下都既可以使用直通网线，又可以使用交叉网线，所以目前一般只做直通网线即可。

表 6-5 100Base-TX 交叉网线引脚分配对比表

引脚号	5 类 UTP 电缆芯线颜色		5 类 STP 电缆芯线颜色	
	一端颜色	另一端颜色	一端颜色	另一端颜色
1	橙白色	绿白色	橙色	红色
2	橙色	绿色	—	—
3	绿白色	橙白色	—	—
4	—	—	—	—
5	—	—	红色	橙色
6	绿色	橙色	黑色	绿色
7	—	—	—	—
8	—	—	—	—
9	N/A	N/A	绿色	黑色
10	N/A	N/A	公共地	公共地

从表6-5可以看出，在使用非屏蔽双绞线（UTP）制作交换网线时两端的芯线呈以下关系：

□一端的1号引脚所对应的芯线与另一端的3号引脚对应的芯线一样，即1号引脚与3号引脚对调；

□一端的2号引脚所对应的芯线与另一端的6号引脚对应的芯线一样，即2号引脚与6号引脚对调。

在使用屏蔽双绞线（STP）制作交换网线时两端的芯线呈以下关系：

□一端的1号引脚所对应的芯线与另一端的5号引脚对应的芯线一样，即1号引脚与5号引脚对调；

□一端的6号引脚所对应的芯线与另一端的9号引脚对应的芯线一样，即6号引脚与9号引脚对调。

2.100Base-T4

100Base-T4是用来向下兼容那些已经安装了3类或4类非屏蔽双绞线电缆（当然也可以使用5类或超5类双绞线）的快速以太网规范，但它要求同时使用双绞线电缆中的全部四对（8根）芯线，这也就是T4的由来。

100Base-T4规范采用半双工交换方式，在它的四对芯线中，三对用于一起发送数据，第四对用于冲突检测（其目的就是为了实现在3类或4类非屏蔽双绞线电缆中实现100Mbps的传输速率）。其RJ—45连接器引脚与双绞线芯线的连接分配如表6-6所示。

表 6-6 100Base-T4 水晶头引脚、信号功能与双绞芯线的连接分配表

引脚号	信号功能	芯线颜色
1	TX_D1+	橙白色
2	TX_D1-	橙色
3	RX_D2+	绿白色
4	BI_D3+	蓝色
5	BI_D3-	蓝白色
6	RX_D2-	绿色
7	BI_D4+	棕白色
8	BI_D4-	棕色

表6-7所示的是100Base-T4规范中的交叉双绞网线RJ-45连接器引脚与双绞芯线的连接分配表。

表 6-7 100Base-T4 交叉双绞网线 RJ-45 连接器引脚与双绞芯线的连接分配表

引脚号	一端芯线颜色	另一端芯线颜色
1	橙白色	绿白色
2	橙色	绿色
3	绿白色	橙白色
4	蓝色	棕白色
5	蓝白色	棕色
6	绿色	橙色
7	棕白色	蓝色
8	棕色	蓝白色

从表6-7中可以看出，在制作100Base-T4规范交叉双绞线时两端的芯线呈以下关系：

□一端的1号引脚所对应的芯线与另一端的3号引脚对应的芯线一样，即1号引脚与3号引脚对调；

□一端的2号引脚所对应的芯线与另一端的6号引脚对应的芯线一样，即2号引脚与6号引脚对调；

□一端的4号引脚所对应的芯线与另一端的7号引脚对应的芯线一样，即4号引脚与7号引脚对调；

□一端的5号引脚所对应的芯线与另一端的8号引脚对应的芯线一样，即5号引脚与8号引脚对调。

在100Base-T4规范中，原来的1、3和2、6这两对双绞芯线与100Base-TX规范一样，仍只能采用半双工传输，但另外两对双绞芯线可以全双工传输，如图6-34所示。

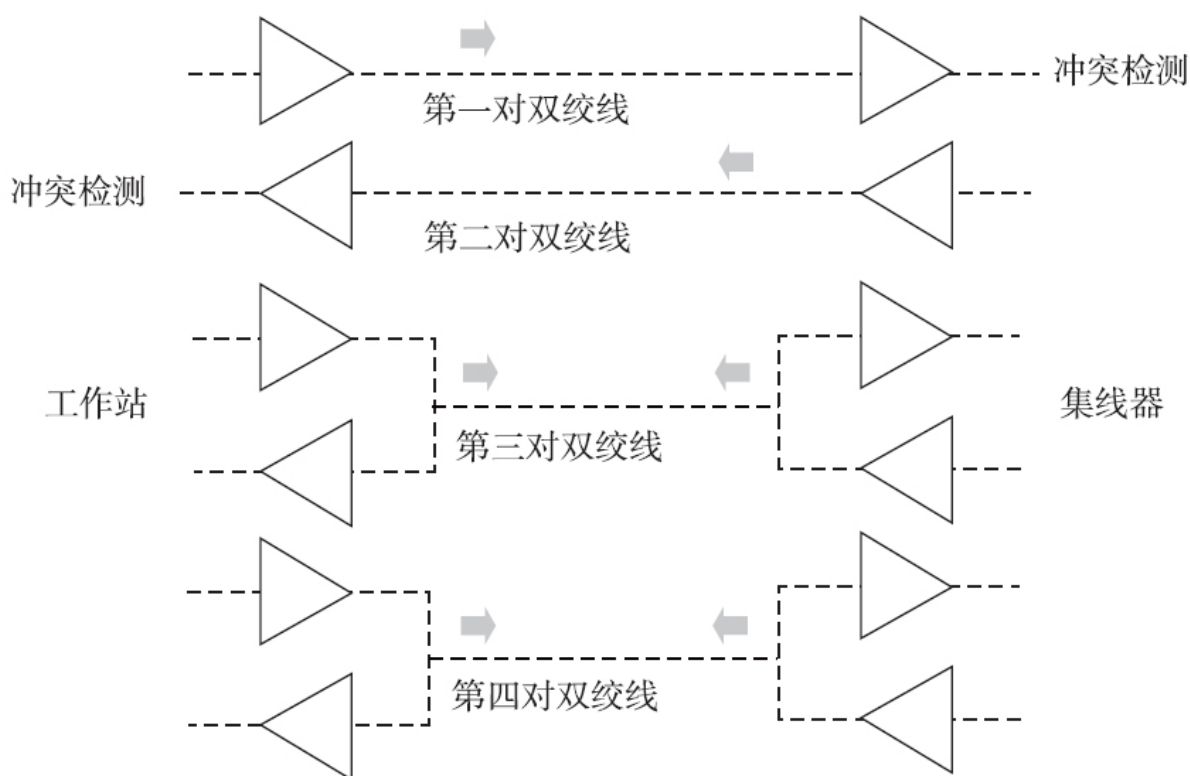


图 6-34 100Base-T4规范中的双绞芯线使用情况

当工作站传送数据给集线器或交换机时，第1、3、4对双绞线用来传送数据，而第2对双绞线则用来检测冲突；而当集线器或交换机传送数据给工作站时，第2、3、4对双绞线用来传送数据，而第1对双绞线则用来检测冲突。也就是说，每一个方向的传送都同时使用了三对双绞线来传送数据。这样一来，对于100Mbps的传送速率来说，每一对双绞线的传送速率只有33.33 Mbps，更加容易实现。另外，100Base-T4规范所采用的是8B/6T编码方法（不再是其他快速以太网规范所使用的4B/5B编码方式），至于这些编码方式在此不做介绍，大家可以查看其他相关资料。

3.100Base-FX

100Base-FX快速以太网规范采用的是两条光纤（单工模式），一条用于发送数据，一条用于接收数据（现在新型的光纤连接方式是仅需一条光纤，可同时用于发送和接收数据，它采用的是波分复用方式）。当工作站的光纤网卡以全双工模式运行时能超过2km。100Base-FX规范中可使用的光纤有以下两类（有关光纤分类的详细知识参见第4章）。

□多模光纤：这种光纤的纤径为62.5/125μm，采用基于LED（发光二极管）的收发器将波长为820nm的光信号发送到光纤上。当连在两个设置为全双工模式的交换机端口之间时，支持的最大距离为2km。

□单模光纤：这种光纤的纤径为 $9/125\mu\text{m}$ ，采用基于激光的收发器将波长为 1300nm 的光信号发送到光纤上。单模光纤率损耗小，较之多模光纤能使光信号传输到更远的距离。

6.6.2 快速以太网物理层结构

快速以太网与前面介绍的标准以太网在物理层结构上存在比较大的区别，具体对比如图6-35所示。下面仍以由低到高的顺序介绍这些逻辑组成部分。

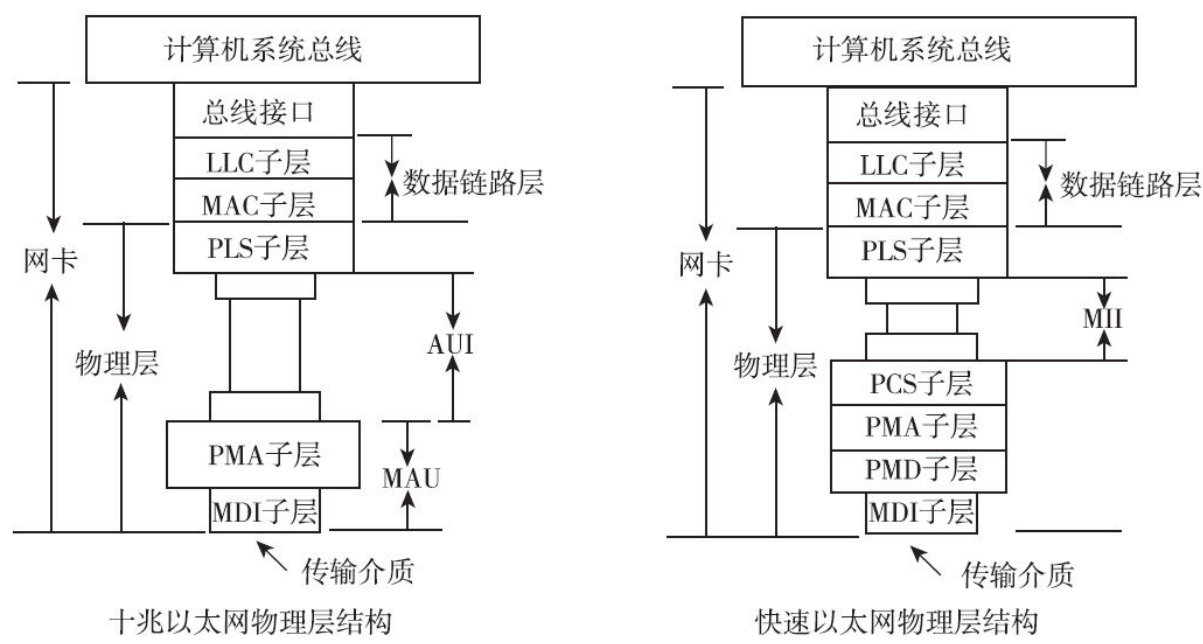


图 6-35 标准以太网与快速以太网的体系结构比较

(1) 介质相关接口子层

在标准以太网物理层结构中的最低层也是介质相关接口（Medium Dependent Interface，MDI）子层，而且功能也是一样的，规定PMD子层和传输介质之间的连接器类型，比如100Base-TX的RJ—45连接器，100Base-FX的SC、ST连接器等。

（2）物理介质相关子层

物理介质相关（Physical Media Dependent, PMD）子层是快速以太网物理层结构中新增加的一层，位于收发器上。它与物理介质直接相连的是信号收发器和信号检测模块，主要提供信号的收发、检测和编/解码等功能。发送信号时，要将来自PMA子层的信号经过适当的编码转换成适合特定传输介质的信号，并提供发送信号驱动；接收信号时，PMD子层经过相应的解码处理后发送给PMA子层。

（3）物理介质连接子层

物理介质连接（Physical Medium Attachment, PMA）子层完成链路监测、载波检测、NRZI（非归零翻转）编/解码、发送时钟合成和接收时钟恢复的功能。

（4）物理编码子层

物理编码子层（Physical Coding Sublayer, PCS）的主要功能是4B/5B编/解码（而不是标准以太网中的曼彻斯特编码，当然在100Base-T4规范中使用的是8B/6T编/解码方式）、碰撞检测和并串转换。

（5）介质无关接口

MII（Medium Independent Interface，介质无关接口）逻辑上与10Mbps以太网的AUI接口对应，可以使MAC子层与传输介质无关。

MII在发送和接收数据时，由原来标准以太网**AUI**的一位位的串行传输改变为半字节（**4**位）的并行传输，这样发送和接收时钟频率只需整个数据传输速率的1/4，即**25MHz**，更容易实现，也更稳定。

（6）协调子层

协调子层（**Reconciliation Sublayer, RS**）替换原来标准以太网**PLS**子层。**PLS**子层的功能是对物理层信号进行编/解码和载波检测，而快速以太网的**RS**是将**MAC**子层的业务定义映射成**MII**接口的信号。

6.7 千兆以太网规范及体系结构

1996年3月IEEE 802委员会成立了IEEE 802.3z工作组，专门负责千兆以太网及其标准，并于1998年6月正式公布关于千兆以太网的标准，后面的IEEE 802.3ab标准下的1000Base-T规范是在1999年6月正式批准发布的，其数据传输率均达到了1000Mbps，即1Gbps，因此又称吉比特以太网。

千兆以太网基本保留了原有以太网的帧结构，同时也支持CSMA/CD介质访问控制技术，所以向下与标准以太网和快速以太网完全兼容，原有的10Mbps以太网或快速以太网可以方便地升级到千兆以太网。

6.7.1 千兆以太网规范

最早在1998和1999年发布的IEEE 802.3z和IEEE 802.3ab标准中就包括了：1000Base-LX、1000Base-SX、1000Base-CX和1000Base-T（前三种统称为1000Base-X子系列）四个，如图6-36所示。其中前三个是由IEEE 802.3z标准规定的，而1000 Base-T规范则是由IEEE 802.3ab标准规定的，是后面专门开发的。这四个千兆以太网规范支持不同类型的光纤和双绞线电缆。

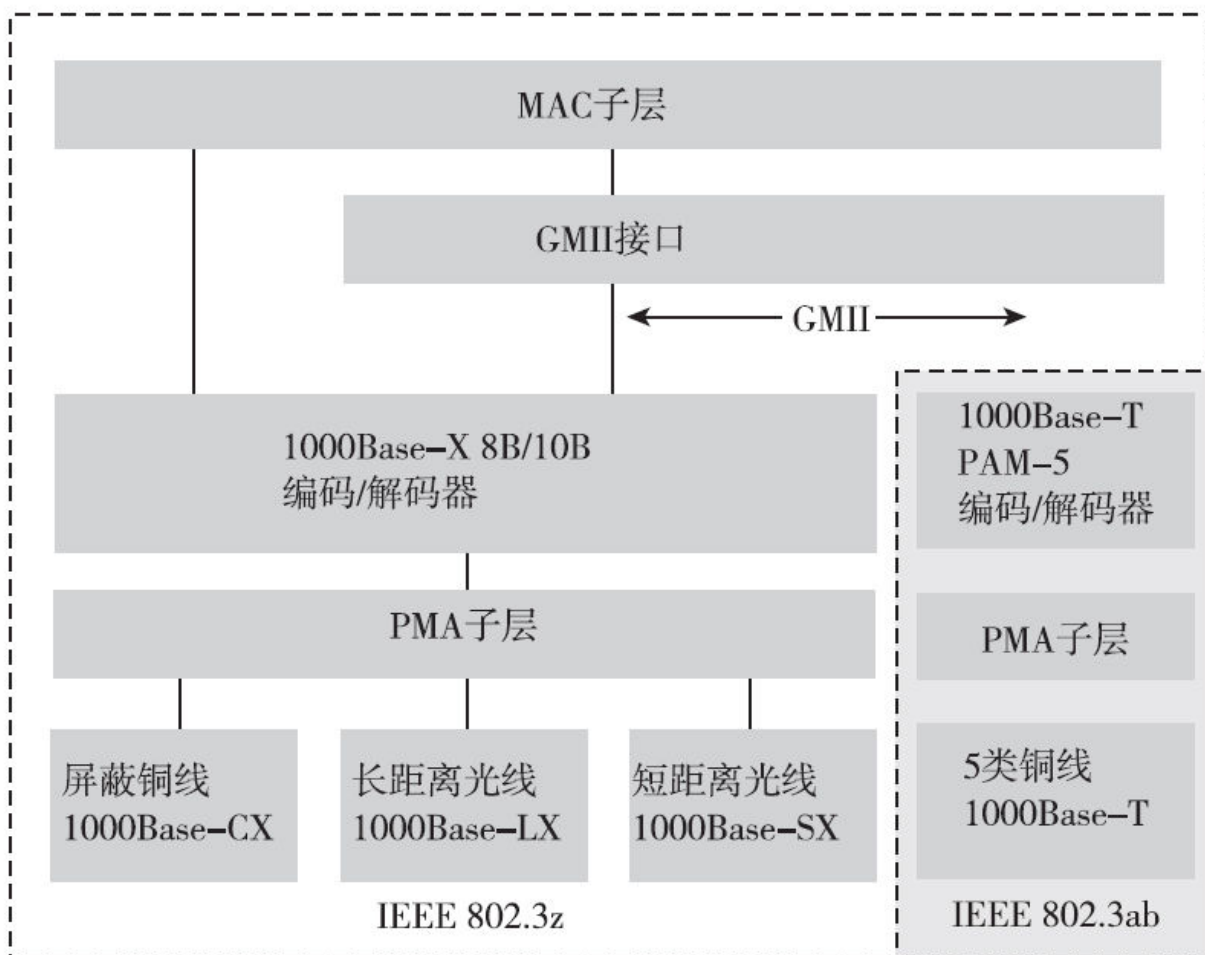


图 6-36 千兆以太网体系结构

除了以上四种以标准形式发布的IEEE千兆以太网规范外，在工业应用中，还有些并没有正式以标准形式对外发布，但却实实在在有广泛应用的千兆以太网规范，如1000Base-LH、1000Base-ZX、1000Base-LX10、1000Base-BX10、1000Base-TX这五种规范。

这样一来，在千兆以太网系列中加起来一共就有九种规范了。在这九种千兆以太网规范中，根据所采用的传输介质类型，总体上是分两大类的：基于光纤的和基于双绞线的。下面分别予以介绍。

1.基于光纤的千兆以太网规范

千兆速率已相当高，从总体性能上来说，最适宜的介质就是光纤了，所以自千兆以太网以后，包括后面的万兆，甚至现在正在研究之中的10万兆以太网规范中，绝大多数是基于光纤这种传输介质开发的。在以上九种千兆以太网规范中，就有六种是基于光纤的。它们分别是已以标准形式发布的1000Base-LX和1000Base-SX，不是以标准形式发布的1000Base-LH、1000Base-ZX、1000Base-LX10和1000Base-BX10。

(1) 1000Base-LX

这是一种通过光纤进行通信的IEEE千兆以太网规范，既可以使用单模光纤（SMF），也可以使用多模光纤（MMF）。1000Base-LX使用长波长激光，波长为1310nm。1000Base-LX规范所使用的光纤主要有线径62.5nm的多模光纤、50nm的多模光纤和9nm的单模光纤。其中使用多模光纤的最大传输距离为550m，使用单模光纤的最大传输距离为5km。1000Base-LX规范采用8B/10B数据编码方法，主要适用于校园网或城域网的主干网。

(2) 1000Base-SX

这也是一种通过光纤进行通信的IEEE千兆以太网规范，适用于线径为50nm和62.5nm的短波（波长为850nm）多模光纤（MMF）。其中

使用62.5nm多模光纤的最大传输距离为275m，使用50nm多模光纤的最大传输距离为550m。1000Base-SX规范也采用8B/10B数据编码方法，适用于大楼网络系统的主干通路。

(3) 1000Base-LH

这是一个非标准的千兆以太网规范，采用的是波长为1300nm或者1310nm的单模或者多模长波光纤维。它类似于1000Base-LX规范，但在单模优质光纤中的最长有效传输距离可达10km，并且可以与1000Base-LX网络保持兼容。

(4) 1000Base-ZX

这也是一个非标准的千兆以太网规范，采用的是波长为1550nm的单模超长波光纤维，最长有效传输距离可达70km。

(5) 1000Base-LX10

这也是一个非标准的千兆以太网规范，采用的是波长为1310nm的单模长波光纤维，最长有效传输距离可达10km。

(6) 1000Base-BX10

这也是一个非标准的千兆以太网规范，其两根光纤所采用的传输介质类型是不同的：下行方向（从网络中心到网络边缘）采用的是波

长为1490nm的单模超长波光纤，而上行方向则是采用1310nm的单模长波光纤，最长有效距离为10km。

2.基于双绞线的千兆以太网规范

采用双绞线作为传输介质的千兆以太网规范有以下三个：

(1) 1000Base-CX

这是一种采用150Ω平衡屏蔽双绞线（STP）作为传输介质（连接器为DB-9）的千兆以太网规范，传输距离最长仅为25m，数据编码方法为8B/10B，适用于数据中心设备间（如交换机之间的连接，尤其适用于主干交换机和主服务器之间的短距离连接），或者堆叠设备间的短距离互连，但不适用于数据中心与配线架的连接。

(2) 1000Base-T

这是一种可以采用5类、超5类、6类或者7类双绞线的全部四对芯线作为传输介质的千兆以太网规范，对应标准为IEEE 802.3ab（与其他千兆以太网标准不同）。它的最大传输距离为100m。在全部的四对双绞芯线中，每对都可以同时进行全双工数据收发，所以即使是相同设备间的连接，也无须制作交叉线，两端都用相同的布线标准即可。这是目前在企业局域网中最常用的一种千兆以太网标准。

(3) 1000Base-TX

在千兆以太网标准中还有一种常用的标准，那就是1000Base-TX，但它不是由IEEE制作的，而是由TIA/EIA于1995年发布的，对应的标准号为TIA/EIA-854。

尽管1000Base-TX也是基于四对双绞线，但却采用快速以太网中100Base-TX标准类似的传输机制，是以两对线发送，两对线接收（类似于100Base-TX的一对线发送，一对线接收）。由于每对线缆本身不同同时进行双向的传输，线缆之间的串扰就大大降低，同时其编码方式也是8B/10B。这种技术对网络的接口要求比较低，不需要非常复杂的电路设计，降低了网络接口的成本。但由于使用线缆的效率降低了（两对线收，两对线发），要达到1000Mbps的传输速率，要求带宽就超过100MHz，也就是说在5类和超5类的系统中不能支持该类型的网络。一定需要6类或者7类双绞线系统的支持。

以上九种千兆以太网规范的比较如表6-8所示，从中可以看出各规范的主要优势和特性。

表 6-8 九种千兆以太网规范比较

千兆以太网规范	使用的传输介质	有效距离
1000Base-CX	150Ω 双绞线	25m
1000Base-LX	波长为 1310nm 的单模或者多模光纤	5km
1000Base-SX	波长为 850nm 的多模光纤	275~550m
1000Base-LH	波长为 1310nm 的单模或者多模光纤	10km
1000Base-ZX	波长为 1550nm 的单模光纤	70km
1000Base-LX10	波长为 1300nm 或 1310nm 的单模光纤或多模光纤	10km
1000Base-BX10	下行波长为 1490nm 的单模光纤，上行波长为 1310nm 的单模光纤	10km
1000Base-T	5 类、超 5 类、6 类或者 7 类双绞线	100m
1000Base-TX	6 类或者 7 类双绞线	100m

6.7.2 1000Base-T以太网技术

在以上九种千兆以太网规范中，性价比最高的就是1000Base-T这种采用普通5类以上双绞线的千兆以太网规范。尽管在双绞线千兆以太网规范中还有一种1000Base-TX的规范，但是它只能使用6类以上的双绞线，网络建设成本明显要高于1000Base-T规范。所以在此专门介绍一下这种应用最广的1000Base-T千兆以太网规范。

1000Base-T是专门为在5类双绞线上进行千兆速率数据传输而设计的。它采用了双绞线的全部四对芯线，并且是全双工传输的，也就是每对双绞线都可以同时进行数据的发送和接收，这样一来1000Mbps的传送速率可以等效地看作在四对双绞线上，每对的传送速率为250Mbps（ $1000\text{Mbps}/4=250\text{Mbps}$ ）。

因为1000Base-T只支持全双工传输，所以与1000Base-T千兆以太网端口直接相连的端口也必须是支持全双工的以太网端口（最佳情况则是同时为1000Base-T千兆以太网端口），而不能是半双工的，否则一方面性能会严重下降，达不到千兆的效果，另一方面还会有严重的丢包现象。相比之下，另一个千兆以太网规范——1000Base-TX尽管也是采用了全部的四对双绞芯线，但是它同一时间是两对（4根）芯线用于发送，两对（4根）芯线用于接收，属于半双工模式，类似于快速以太

网中的100Base-TX规范，只不过100Base-TX规范中，同一时间发送数据和接收数据都仅用一对（2根）芯线。

1000Base-TX和1000Base-T规范双绞芯线使用情况的比较如图6-37所示（注意区分图中的不同颜色）。1000Base-T规范中各芯线的具体作用如图6-38所示，每条芯线上分担的速率都是250Mbps，所以在全双工模式下，它可以实现2Gbps的传输速率。这得益于1000Base-T采用了一种更加强大的信号传输和编/解码方案——PAM-5（Pulse Amplitude Modulation 5，脉冲调幅-5）。PAM-5编码使用-2V、-1V、0V、+1V、+2V五种电平，其中-2V、-1V、+1V、+2V这四种电平用于信号编码，0电平用于前向纠错编码（FEC）。而在100Base-TX中采用的是3级MLT（3Multi-Level Transmission，多级传送）编码方案MLT-3，使用-1V、0V、+1V三种电平，其中-1V、+1V这两种电平用于信号编码，0电平用于前向纠错编码。由此可见，PAM-5编码方案在链路上较MLT-3编码方案可以多传送一倍的数据。

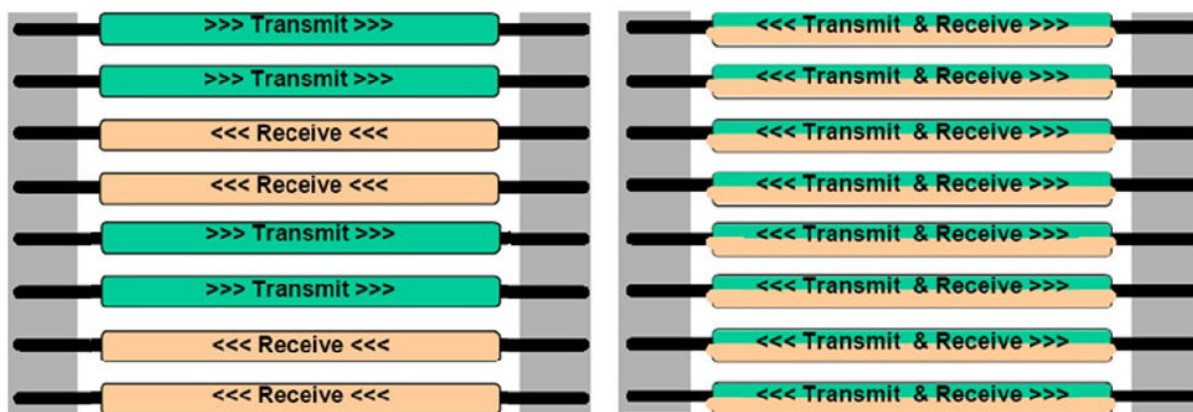


图 6-37 100Base-TX和100Base-T规范中双绞芯线使用比较

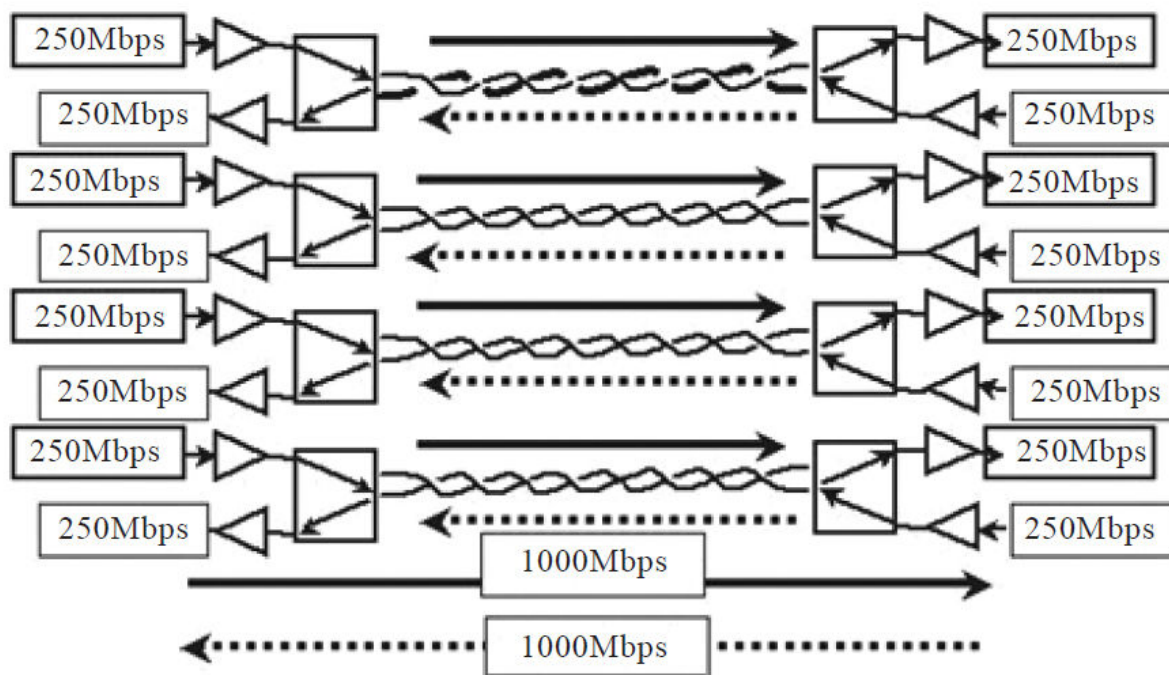


图 6-38 100Base-T规范中各双绞芯线的作用

还有一个要注意的地方就是，1000Base-T规范中不采用交叉网线，只需直通线（两端按同一标准制作网线），因为这一规范中总是采取全双工传输的，任何一根网线都可以同时发送和接收数据。这与1000Base-TX是不一样的（有X的表示是不同设备连接时要采用交叉线，而没有X的表示为不用交叉线）。

总体而言，1000Base-T规范最吸引人的地方在于为企业提供了一种除多模光纤以太网方案外的更廉价的千兆方案，用户可以在原来100Base-T的基础上进行平滑升级到1000Base-T。该规范主要用于结构化布线中同一层建筑间的通信，可以利用现有以太网或快速以太网已

铺设的UTP电缆进行网络升级，其可被用做大楼内的网络主干，大大节省了成本。这是目前最主要应用的千兆以太网方案。

6.7.3 IEEE千兆以太网物理层结构

尽管在以上九种千兆以太网中有些并不是IEEE发布的，但总体上来说，他们的体系结构与IEEE发布的千兆以太网规范类似，所以在此仅以IEEE发布的千兆以太网标准为例介绍其体系结构。

与本章前面介绍的标准以太网、快速以太网物理层结构一样，千兆以太网的整个体系结构差别主要体现在物理层和MAC子层，LLC子层基本上都是保持了一致。从体系结构上来看，千兆以太网与快速以太网的区别并不是很大，主要就是从原来的MII接口替换成了GMII接口。图6-39所示的是快速以太网体系结构与千兆以太网体系结构的比较。

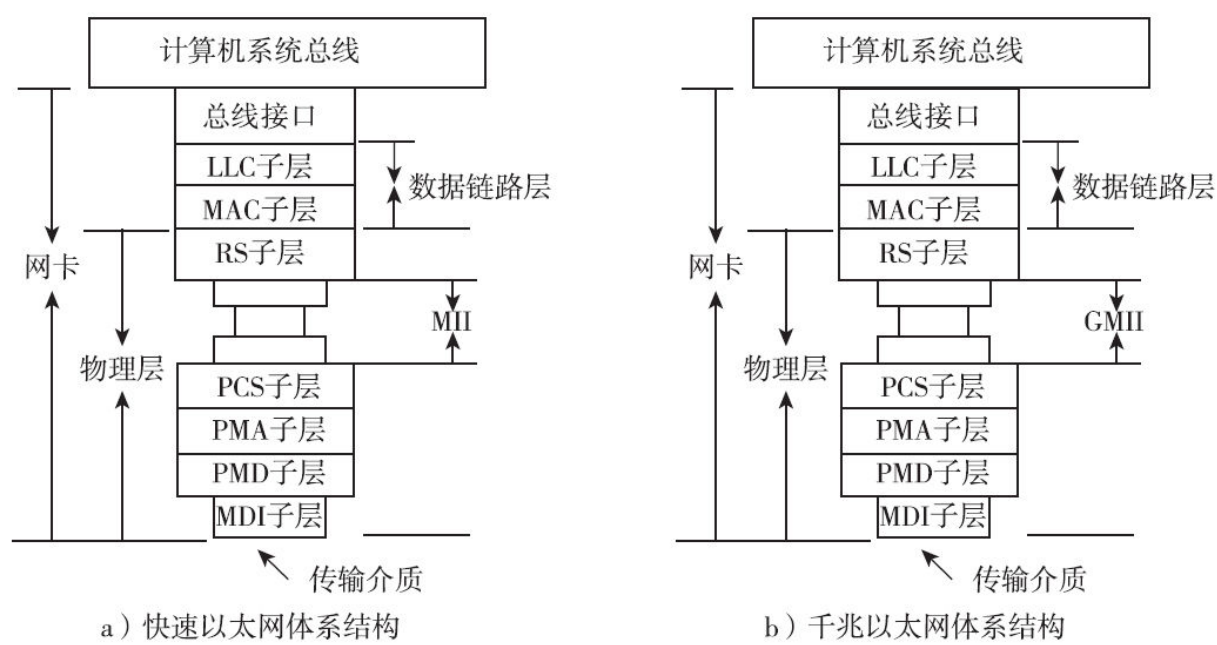


图 6-39 快速以太网与千兆以太网体系结构的比较

快速以太网体系结构与千兆以太网体系结构在物理层上的主要区别表现为RS子层和PCS子层之间的接口发生了改变：由快速以太网中的介质无关接口（MII）扩展为千兆介质无关接口（GMII）。其他各子层的功能参见本章前面介绍的快速以太网标准中的物理层结构。

GMII的发送和接收数据宽度由原来的MII的半字节（4位）扩展为一字节（8位），这样每根芯线使用125MHz的时钟频率可以实现1000Mbps的传输速率。另外与MII、AUI不同，GMII不支持连接器和电缆，其只是内置作为IC和IC之间的接口。

另外，千兆以太网物理层的PCS子层与快速以太网中的PCS子层有所区别，而且1000Base-X子系列三个规范物理层与1000Base-T规范物理层中的PCS子层也不一样，前者采用的是8B/10B编码方式，而后者采用的是PAM-5编码方式。快速以太网100Base-TX规范中采用的是4B/5B编码方式。

6.8 万兆以太网规范及体系结构

1999年底成立了IEEE 802.3ae工作组专门进行万兆以太网（Ten Gigabit Ethernet）技术（10Gbps）的研究，并于2002年以IEEE 802.3ae标准的形式第一次发布万兆以太网标准，这个标准是当时最快的以太网标准。在10Gbps以太网中，最显著的改变就是只能与全双工交换机连接，不再支持半双工连接，也不支持CSMA/CD。

6.8.1 万兆以太网规范

万兆以太网标准和规范都比较多：在标准方面，有2002年的IEEE 802.3ae，2004年的IEEE 802.3ak，2006年的IEEE 802.3an和IEEE 802.3aq，以及2007年的IEEE 802.3ap。

在万兆以太网规范方面，仅由上述IEEE标准中发布的规范就有10多个，如2002年在IEEE 802.3ae标准中发布的基于光纤的规范包括：10GBase-SR、10GBase-LR、10GBase-ER、10GBase-LX4、10GBase-SW、10GBase-LW、10GBase-EW；2004年在IEEE 802.3ak标准中发布的基于双绞线的10GBase-CX4；2006年在IEEE 802.3an标准发布的基于双绞铜线的10GBase-T；2006年在IEEE 802.3aq标准中发布的基于光纤的10GBase-LRM；2007年在IEEE 802.3ap标准中发布的基于铜线的

10GBase-KR和10GBase-KX4。除此之外，还有一些不是由IEEE发布的万兆以太网规范，如Cisco的10GBase-ZR和10GBase-ZW，这些规范所对应的标准具体如图6-40所示。

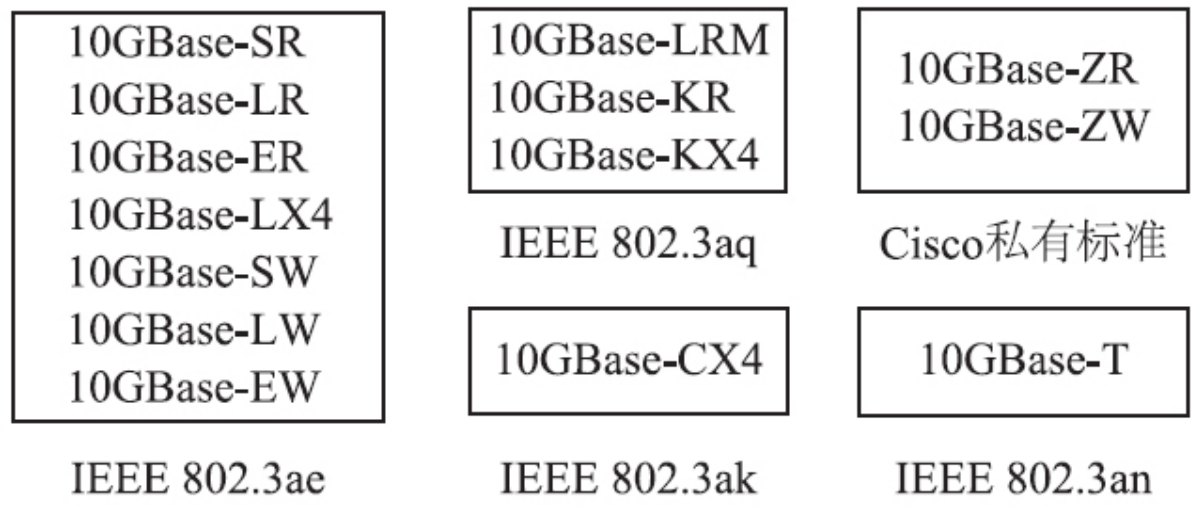


图 6-40 万兆以太网标准与对应的规范

以上这10多种万兆以太网规范可以分为三类：一是基于光纤的局域万兆以太网规范，二是基于双绞线（或铜线）的局域万兆以太网规范，三是基于光纤的广域万兆以太网规范。下面分别予以介绍。

1.基于光纤的局域万兆以太网规范

用于局域网的光纤万兆以太网规范有：10GBase-SR、10GBase-LR、10GBase-LRM、10GBase-ER、10GBase-ZR和10GBase-LX4这六个。

(1) 10GBase-SR

10GBase-SR中的SR是Short Range（短距离）的缩写，表示仅用于短距离连接。该规范支持编码方式为64B/66B的短波（波长为850nm）多模光纤（MMF），有效传输距离为2~300m。但要支持300m传输需要采用经过优化的50 μ m线径OM3（Optimized Multimode 3，优化的多模3）光纤（没有优化的线径50 μ m光纤称为OM2光纤，而线径为62.5 μ m的光纤称为OM1光纤）。

（2）10GBase-LR

10GBase-LR中的LR是Long Range（长距离）的缩写，表示主要用于长距离连接。该规范支持编码方式为64B/66B的长波（1310nm）单模光纤（SMF），有效传输距离为2m~10km，事实上最高可达到25km。

（3）10GBase-LRM

10GBase-LRM中的LRM是Long Reach Multimode（长距离延伸多点模式）的缩写，表示主要用于长距离的多点连接模式，对应的标准为2006年发布的IEEE 802.3aq，采用64B/66B编码方式。采用该规范时，在1990年以前安装的线径62.5 μ m多模光纤FDDI网络和100Base-FX网络中的有效传输距离为220m，而在OM3光纤中可达260m，在连接长度方面，不如以前的10GBase-LX4规范，但是它的光纤模块比10GBase-LX4规范光纤模块具有更低的成本和更低的电源消耗。

(4) 10GBase-ER

10GBase-ER中的ER是Extended Range（超长距离）的缩写，表示连接距离可以非常长。该规范支持编码方式为64B/66B的超长波（1550nm）单模光纤（SMF），有效传输距离为（2~40）km。

(5) 10GBase-ZR

10GBase-ZR中的ZR是Ze best Range（最长距离）的缩写，表示连接距离最长，可达80km，是Cisco的一个私有万兆以太网标准。它使用的也是超长波（1550nm）单模光纤（SMF）。

(6) 10GBase-LX4

10GBase-LX4规范在IEEE 802.3ae标准中发布，设计通过波分复用技术采用4束光波通过单对光学电缆来发送信号，采用8B/10B编码方式。10GBase-LX4工作波长为1310nm，使用多模或单模暗光纤，主要适用于需要在一个光纤模块中同时支持多模和单模光纤的环境。多模光纤传输距离为240~300m，单模光纤10km以上，根据电缆类型和质量，还能达到更远的距离。

2.基于双绞线（或铜线）的局域网万兆以太网规范

在2002年发布的几个万兆以太网规范中并没有支持铜线这种廉价传输介质的，但事实上，像双绞线这类铜线在局域网中的应用是最普

遍的，其不仅成本低，还容易维护，所以在近几年相继推出了多个基于双绞线（6类以上）的万兆以太网规范。它们包括10GBase-CX4、10GBase-KX4、10GBase-KR、10GBase-T。下面分别予以简单介绍。

（1）10GBase-CX4

10GBase-CX4规范使用IEEE 802.3ae中定义的XAUI（万兆附加单元接口）和用于InfiniBand中的4X连接器，传输介质称为CX4铜缆（其实就是一种屏蔽双绞线）。它的有效传输距离仅为15m。

10GBase-CX4规范不是利用单个铜线链路传送万兆数据，而是使用4台发送器和4台接收器来传送万兆数据，并以差分方式运行在同轴电缆上，每台设备利用8B/10B编码，以每信道3.125GHz的频带传送2.5Gbps的数据。这需要在每条电缆组的总共8条双同轴信道的每个方向上有4组差分线缆对。另外，与可在现场端接的5类、超5类双绞线不同，CX4的线缆需要在工厂端接，因此客户必须指定线缆长度。线缆越长一般直径就越大。

（2）10GBase-KX4和10GBase-KR

10GBase-KX4和10GBase-KR两个规范主要用于设备背板连接中，如刀片服务器、路由器和交换机的集群线路卡，所以又称背板以太网。

万兆背板连接目前已经存在并行和串行两种版本：并行版本（10Gbase-KX4规范）是背板的通用设计，将万兆信号拆分为4条通道（类似XAUI），每条通道的带宽都是3.125Gbps。串行版本（10GBase-KR规范）中只定义了一条通道，采用64/66B编码方式实现10Gbps高速传输。10Gbase-KX4使用与10GBase-CX4规范一样的物理层8B/10B编码，10GBase-KR使用与10GBase-LR/ER/SR这3个规范一样的物理层64B/66B编码。

（3）10GBase-T

10GBase-T是基于屏蔽或非屏蔽双绞线，主要用于局域网的万兆以太网规范，最长传输距离为100m。这可以算是万兆以太网一项革命性的进步，因为在此之前，一直认为在双绞线上不可能实现这么高的传输速率，原因就是运行在这么高工作频率（至少为500MHz）基础上损耗太大。但标准制定者依靠以下4项技术构件使10GBase-T变为现实：损耗消除、模拟到数字转换、线缆增强和编码改进。

在编码方面，10GBase-T不是采用原来1000Base-T的PAM-5编码方式，而是采用PAM-8编码方式，支持833Mbps和400MHz带宽，对布线系统的带宽要求也相应地修改为500MHz。如果仍采用PAM-5的10GBase-T，对布线带宽的需求是625MHz。在连接器方面，10GBase-T使用已广泛应用于以太网的650 MHz版本RJ-45连接器。在6类线上最长有效传输距离为55m，而在6a类线上可以达到100m。

3.基于光纤的广域网万兆以太网规范

10Gbps以太网一个最大改变就是它不仅可以在局域网中使用，还可应用于广域网中，其对应的规范包括10GBase-SW、10GBase-LW、10GBase-EW和10GBase-ZW（此为Cisco私有标准）。这4个广域网10Gbps广域以太网规范专为工作在OC-192/STM-64 SDH/SONET环境而设置的，使用SDH（Synchronous Digital Hierarchy，同步数字体系）/SONET（Synchronous optical networking，同步光纤网络）帧，传输速率为9.953 Gbps。它们所使用的光纤类型和有效传输距离分别对应于前面介绍的应用于局域网中的10GBase-SR、10GBase-LR、10GBase-ER和10GBase-ZR规范。

表6-9所示综合了以上介绍的所有10Gbps以太网规范，在实际的网络系统设计中，就可以针对具体的节点环境和网络需求来对应选择了。当然，目前10Gbps以太网主要是在骨干网络中采用，在一般的企业局域网中仍很少见到。

表 6-9 万兆以太网规范比较

万兆以太网规范	使用的传输介质	有效距离	应用领域
10GBase-SR	850nm 多模光纤, 50μm 的 OM3 光纤	300m	局域网
10GBase-LR	1310nm 单模光纤	10km	
10GBase-LRM	62.5 μm 多模光纤, OM3 光纤	260m	
10GBase-ER	1550nm 单模光纤	40km	
10GBase-ZR	1550nm 单模光纤	80km	
10GBase-LX4	1300nm 单模或者多模光纤	300m (多模时), 10km (单模时)	
10GBase-CX4	屏蔽双绞线	15m	
10GBase-T	6 类、6a 类双绞线	55m (6 类线时), 100m (6a 类线时)	背板以太网
10GBase-KX4	铜线 (并行接口)	1m	
10GBase-KR	铜线 (串行接口)	1m	SDH/SONET 广域网
10GBase-SW	850nm 多模光纤, 50μm 的 OM3 光纤	300m	
10GBase-LW	1310nm 单模光纤	10km	
10GBase-EW	1550nm 单模光纤	40km	
10GBase-ZW	1550nm 单模光纤	80km	

6.8.2 万兆以太网的物理层结构

在许多万兆以太网规范中，也对应了许多不同类型的万兆以太网物理层，但总体类型还是与最初于2002年发布的几类万兆以太网规范差不多。下面分基于光纤传输介质万兆以太网规范物理层和基于铜线传输介质万兆以太网规范物理层两种类型进行介绍。

在IEEE发布的万兆以太网规范中，主要是三大类，即10GBase-X（仅包括10GBase-X规范）、10GBase-R（包括10GBase-SR、10GBase-LR和10GBase-ER等3个规范）和10GBase-W（包括10GBase-SW、10GBase-LW和10GBase-EW等3个规范）。这三大类规范所对应的体系结构如图6-41所示（注意其中用深颜色标注的部分）。

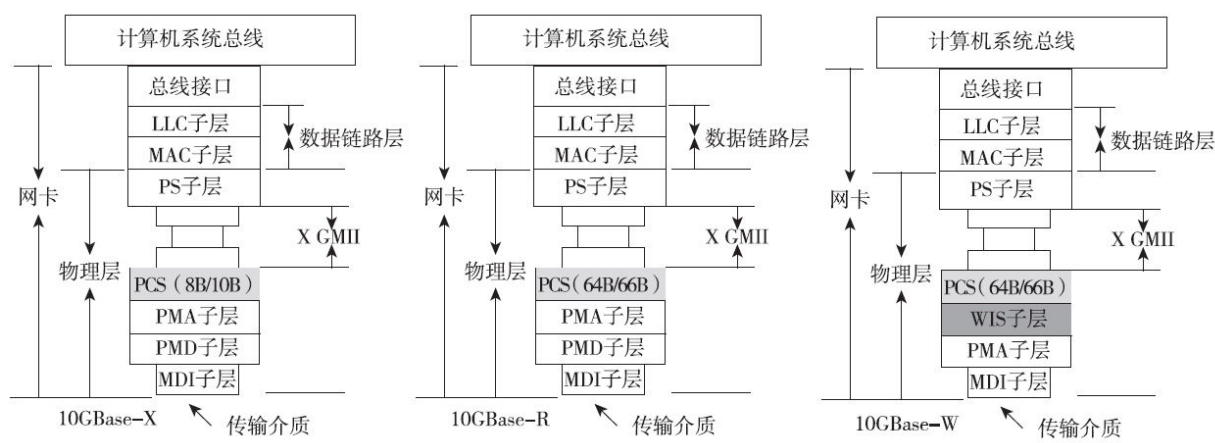


图 6-41 三个万兆以太网规范子系列的体系结构

通过与图6-39所示的千兆以太网体系结构比较，从层次上来看，主要区别如下：

1) 把原来千兆以太网的GMII升级替换成了XGMII（10 Gigabit Media Independent Interface，10G介质无关接口）。

XGMII与千兆以太网的GMII一样，也是位于收发器内部，但它是由全新的芯片完成的，以实现支持10Gbps传输速率。XGMII功能由一个具有74条信号线（相当于74bit宽度）的接口芯片完成，其中的64条数据线用于数据的收发（收、发各32条），其余的为时钟和控制信号位，是连接以太网MAC子层和物理层的桥梁。

2) 在10GBase-R子系列中的3个规范中的物理层，除了上述接口换成为XGMII外，还有一个区别就是PCS子层的编码方式由原来的8B/10B改变成了64B/66B。

3) 在10GBase-W子系列中的3个规范中相对千兆以太网物理层的改变更大，除了在10GBase-R子系列中的两处改变外，还在PCS子层与PMA子层之间增加了一个新的子层——WIS（WAN Interface Sublayer，WAN接口）子层。

WIS子层用于在广域网中产生适配ANSI定义的SONET STS-192c传输格式，或适配ITU定义SDH VC-4-64c容器速率的以太网数据流，这

就使万兆以太网设备与同步光纤网络（SONET）STS-192c传输格式相互兼容。

其他各子层的功能与千兆以太网体系结构中的对应的各子层类似，可参见对应各子层的功能说明，只不过在PCS子层中所采用的编码与千兆以太网不一样，参见图6-39所示。

6.9 IEEE 802.1d协议

IEEE 802.1d也就是我们通常所说的STP（Spanning Tree Protocol，生成树协议）。其设计目的就是要使交换网络中没有二层环路。在这里首先我们介绍什么是环路。

6.9.1 理解“网络环路”

这里所说的环路就是指网络环路。它又分为二层环路和三层环路两种。二层环路是指在二层交换网络中，网络广播信息在网络中形成的一个封闭的环路。如图6-42所示，SW1和SW2这两个交换机之间有两条链路连接。现假设SW1的F0/3端口收到了来自PC1的一个广播包，它会通过F0/1和F0/2端口分别向SW2的F0/1和F0/2端口发送，同样当SW2的F0/1和F0/2端口在收到广播包后也会分别再向SW1的F0/1和F0/2端口发送广播包，以此反复就形成了一个恶性死循环，即形成广播风暴，造成了大量网络设备和带宽资源的浪费。

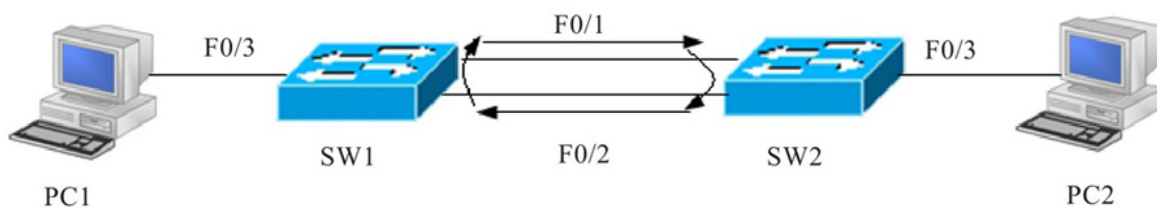


图 6-42 冗余链路二层环路示例

上面所说的是广播包在二层环路中的传播情形，其实即使是从PC1发送到PC2的单播包，因为SW1和SW2有两条链路，这样在SW2上可能会同时收到来自SW1的F0/1和F0/2端口发来的包，最终造成重复包，也浪费了网络设备和带宽资源。在如图6-43所示的封闭环交换网络中同样存在这样的情况，大家可以自己分析一下。

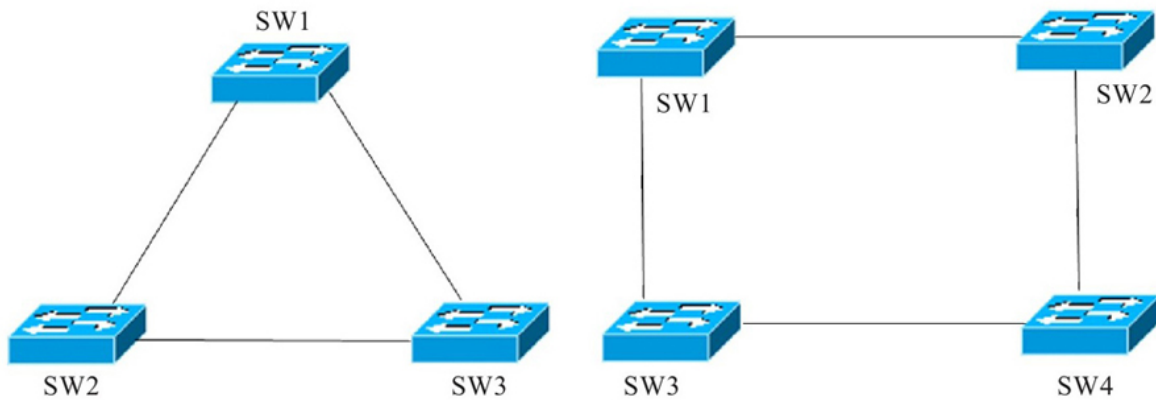


图 6-43 封闭的二层环路示例

而三层环路则是指当原路由意外不能工作时，造成路由通告错误所形成的一个恶性路由循环。当然主要在RIP、IGRP这两种动态路由协议中才会出现这种情况。

6.9.2 STP简介

“广播风暴”的现象只存在于两点之间存在冗余链路（也就是从一个节点到达另一节点存在多条路径）的网络之中。冗余链路在现实网络设计中大量存在。如图6-44所示，核心层、汇聚层的交换机间都是有冗余设计的。

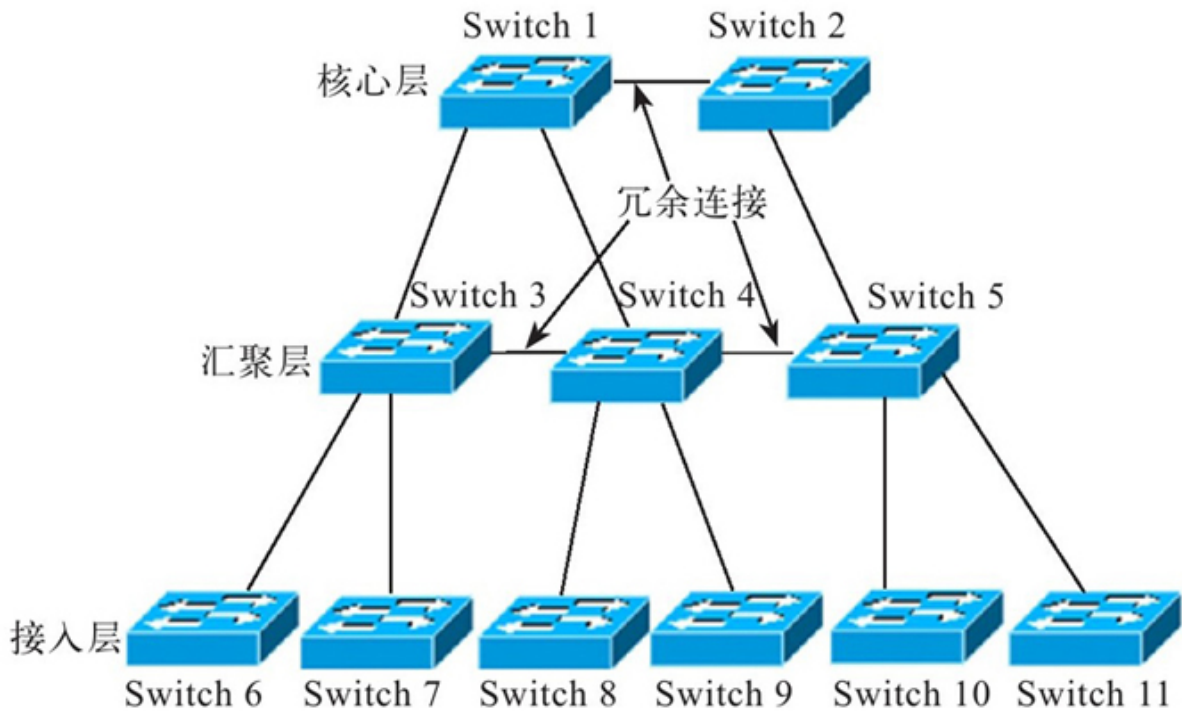


图 6-44 冗余连接设计示例

本来这样设计的目的是想当某一条链路失效时，另一条冗余的链路能够马上接管所有的工作（如HSRP、VRRP协议的热备应用）。但是，在实际工作中这些冗余连接还会带来广播风暴隐患。如从图6-44中

的Switch 6交换机发送的数据要到达Switch 1，可以有多条不同的路径，其中包括Switch 6→Switch 3→Switch 1、Switch 6→Switch 3→Switch 4→Switch 1、Switch 6→Switch 3→Switch 4→Switch 5→Switch 2→Switch 1这3条。这样一来，当从Switch 6发送一个请求数据包到Switch 1时，就可能从以上3条路径中同时流出，Switch 1交换机上可能会收到多个重复的包，这显然是不需要的。本示例只是一个很简单的结构，如果结构更复杂的话，冗余连接会更多，对应的冗余链路也就更多，所带来的广播风暴将更大。

除了广播风暴的影响外，这么多冗余链路，还可能形成网络数据发送和接收死循环。同样以图6-40为例。在以上3条路径中，从Switch 1上发送到Switch 6的数据包，可能就不会到达Switch 6，而是直接在Switch 1→Switch 3→Switch 4→Switch 1、Switch 1→Switch 4→Switch 3→Switch 1、Switch 1→Switch 2→Switch 5→Switch 4→Switch 1等多个封闭的环路上循环。

为了解决广播风暴和网络死循环这两个在二层数据网络中存在的主要弊端，IEEE（电气和电子工程师学会）在基于思科早期开发的生成树技术基础上发布了IEEE 802.1d协议标准，也就是我们通常所说的STP。

STP是一个确保整个交换网络无环路拓扑结构的OSI二层协议。它允许在网络设计中包括冗余链路，以提供在主链路失效时自动接替其

工作的备份链路，同时又不会生成二层环路。必须避免二层环路，否则可能导致数据包在网络中循环传输。**STP**就像其名称一样，它在一个由网桥或交换机连接的网状网络内部创建一棵生成树，禁止不属于树的那些链路，使得任意两个网络节点间仅有一条活动的路径。**STP**将两点之间存在的多条路径划分为通信链路和备份链路，并规定数据的转发只在通信链路上进行，而备份链路只用于链路的侦听，只有在发现通信链路径失效时，才自动将通信切换到备份链路上。

用于正常数据转发的“通信链路”往往是最短的路径，如图6-44所示示例中的Switch 6→Switch 3→Switch 1这条路径。但是，**STP**并不是全部禁用其余冗余链路，而只是在正常工作中让它们处于侦听状态，当活动链路失效时才接替活跃链路的工作。这样就可以达到既不会因为冗余链路而形成广播风暴，也不会因为没有冗余链路而造成网络循环，从而提高了网络的可用性。

6.9.3 STP的基本工作原理

IEEE 802.1d STP是把整个交换网络看成一个生成树实例（也就是一棵“树”），称之为CST（Common Spanning Tree，公共生成树）。在这里，我们先要理解为什么要以“树”来形容STP所形成的拓扑结构。

我们知道，自然界的树是由树根、树干、树枝和树叶组成的。我们可以把树叶比作网络用户的主机或服务器（连接在接入层交换机的端口上），把树枝比作网络中接入层交换机（连接在汇聚层交换机的端口上），把树干比作网络中汇聚层交换机（连接在核心层交换机的端口上），把树根比作网络中核心交换机，如图6-45所示。从图中可以看出，网络中的任何节点（端口）之间的连接都不存在环路，因为任意两个节点间的连接只有一条路径，也就是我们通常所说的“单线连接”。STP的目的就是要将一个物理网络拓扑结构变成这种无环路的逻辑拓扑结构。

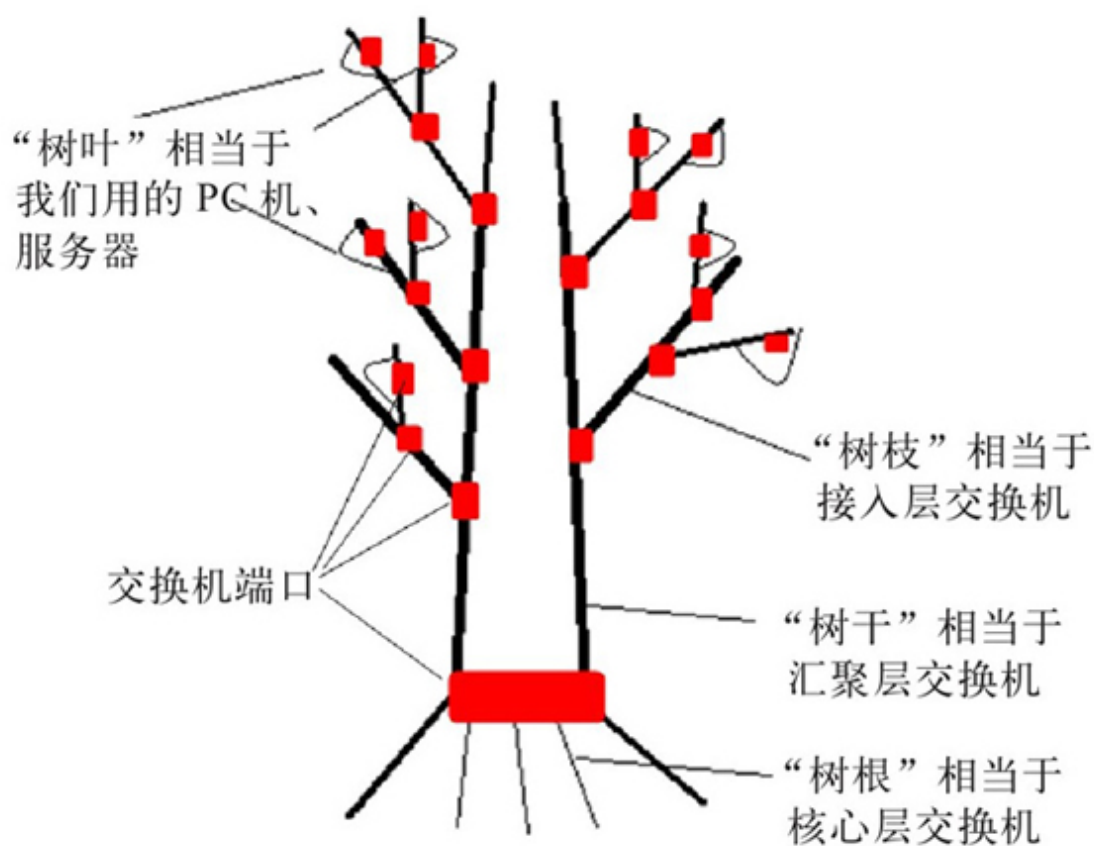


图 6-45 STP生成树模拟示例

在如图6-42所示的示例中，在SW1和SW2之间存在一条冗余链路。从PC1发出的一个广播或者多播包可能会在两个交换机之间循环，这在前面已作了分析。但是，如果在两个交换机上运行了STP，则其中一条链路会自动断开。这时在SW1和SW2之间就只有一条路径，从PC1发出的广播或者多播包就不会在两个交换机之间循环了，消除了网络环路。

那么STP又是如何做到在有冗余链路的情况下同时避免环路发生的呢？也就是STP如何把一个有环路的物理交换网络转换成逻辑意义上无

环路的树形交换网络呢？其实原理很简单，它采用两项主要技术：一是通过选举，使得交换网络中的各个交换机处于特定的角色，就像图6-45中的各交换机分别为位于树根的核心交换机，位于树干的汇聚交换机，位于树枝的接入交换机，这样就可以使整个交换网络拓扑结构呈一个树形结构；另外一项技术就是把交换机间连接的端口（连接主机的端口不算在内）划分为不同角色，各个交换机端口又可设定成几种特定状态，使可能形成环路的一些交换机端口在正常工作时处于阻塞状态，同时指定每个交换机与上级交换机连接的固定端口。

在交换机角色选举中，也就是在整个交换网络中选举出一个根交换机（网桥时代称之为根桥，**Root Bridge**），所连接的每个网段的交换机为指定交换机（**Designated Bridge**），或者称之为非根交换机。根交换机一般是位于核心层（需要经过选举确定），相当于树根，主要用于为下面各个网段交换机提供服务，以及作为整个网络的出口。然后又为各交换机定义了根端口（**Root Port**）和指定端口（**Designated Port**）两种主要的端口角色。根端口就是非根交换机到达根交换机的最短路径（也需要选举确定）的那个端口，指定下级交换机要访问根交换机时必须从这个端口出发，而不能从其他端口，所以根端口仅位于非根交换机上。指定端口则有两种情况：根交换机上的所有连接交换机的端口都可以是指定端口，因为它们直接在根交换机上，每个用于连接交换机的端口都连接了一个网段；指定交换机上的指定端口是指与下级交换机根端口连接的那个端口。每个指定交换机只有一个根端

口，但可以有多个指定端口，因为一个交换机可以连接多个网段（每个交换机端口可以连接一个网段），但每个网段只能一个指定端口。

同时，STP中包括的端口状态有：阻塞（**Blocking**）状态、侦听（**Listening**）状态、学习（**Learning**）状态、转发（**Forwarding**）状态、禁止（**Disabled**）状态。在交换机上所有交换机间连接的端口中只有根端口和指定端口是活跃（非“禁止”状态）的，其他交换机间连接的端口都是呈禁止状态。数据的转发路径就是：由下级非根交换机的指定端口到上级非根交换机的根端口，一直到根交换机的指定端口。这样就可避免二层环路的发生。

在图6-44所示的网络中，各交换机角色以及端口角色如图6-46所示，当然这些都是需要经过选举的，具体选举原理大家可以参见笔者编著的《Cisco/H3C交换机配置与管理完全手册》（第2版）一书。

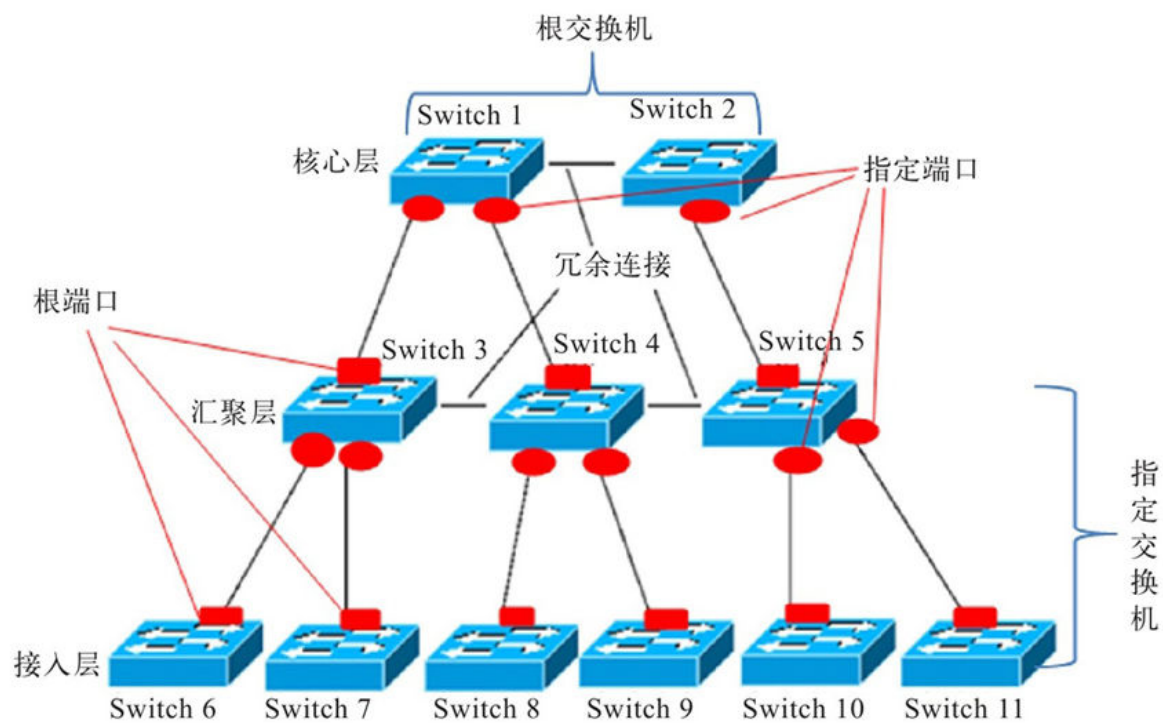


图 6-46 STP交换机和端口角色示例

6.9.4 STP的不足和增强技术

STP协议的算法广泛运用于二层以太网的收敛（收敛是指从网络上拓扑结构有了变化开始到整个网络设备中的信息重新一致为止的整个过程，这个过程所用时间称为收敛时间），但是由于它是局域网的初期开发的技术，所以它也存在着以下几个不足。

（1）二层数据网的收敛时间过长

根据IEEE 802.1d协议的算法，每个节点的初始化时间约为30s，整个拓扑的收敛将会在30~50s之间，即使是一个以太网端口插入计算机也需要这个过程。而我们知道在关键网络中，如主机核心机房的连接，用户期望的值往往要短得多。

（2）网络拓扑容易引起全局波动

由于IEEE 802.1d协议中没有域的概念，网络中用户增加或减少设备、设备配置的改变往往会引起全局不必要的波动。用户如果改变其设备参数甚至可能引起根交换机的改变，从而造成通信网络的中断，这使得用户在大规模的数据网络中不敢轻易使用IEEE 802.1d协议的算法。

（3）缺乏对现有多VLAN环境的支持

IEEE 802.1d协议是针对整个交换网络而言的，也就是整个交换网络就是一个生成树实例（称为单生成树，SST），没有考虑存在多个VLAN的情形。这样一方面不便于基于VLAN来优化整个交换网络拓扑结构，另一方面不便于在复杂网络中消除环路的实现。

针对以上这些STP协议自身的不足，网络设备制造商开发了许多增强的生成树技术，如提高收敛效率的Rapid-STP（RSTP），Cisco基于一个VLAN的PVST（每VLAN生成树）、PVST+、Rapid-PVST+，基于多个VLAN的多生成树MSTP等。具体大家可以参见笔者编著的《Cisco/H3C交换机配置与管理完全手册》（第2版）一书。

6.10 IEEE 802.1q协议

VLAN（Virtual Local Area Network，虚拟局域网）对应的技术标准是IEEE 802.1q，是1999年6月由IEEE委员会正式颁布实施的，而最早的VLAN技术是在1996年由Cisco（思科）公司提出的。随着十多年来的发展，VLAN技术得到广泛的支持，在大大小小的企业网络中广泛应用，成为当前最为热门的一种以太网技术。VLAN属于OSI/RM二层技术，在同一VLAN内部是通过数据链路层（OSI/RM第二层）进行通信的。

6.10.1 划分VLAN的目的

在早期的以太网中，并没有VLAN技术，所以在图6-24中所列的以太网帧格式中没有添加用于标记帧的VLAN信息的字段。那后来又是什么原因开发了这种VLAN技术呢？

说到这里，我们首先要知道，VLAN的主要用途是什么。

VLAN的主要用途就是把一个大的交换网络划分成多个小的交换网络。那为什么要划分成小的交换网络呢？原来，VLAN可以隔离二层通信，也就是不同VLAN中的主机是不能直接进行二层通信的。这样就可以达到一个不划分VLAN的交换网络中不能达到的二层主机隔

离的目的，因为在没有采用VLAN技术的交换网络中，各主机都是可以直接进行二层通信的。但这仍不是划分VLAN的主要目的。

那隔离二层通信的真正目的是什么呢？那就是想缩小广播域，减小二层网络中广播流量对整个交换网络的影响。我们已经知道，在交换网络中，如果在交换机缓存ARP表项中找不到帧中的目的MAC地址，就会在整个交换网络中进行广播。由于交换机的缓存容量有限，加上ARP表项也有保存时间，所以在较大交换网络中，广播可能是经常要发生的，可以说是不可避免的。

同时我们又知道，广播通信只在本网段内进行，不能通过路由器，或者其他三层设备跨网段进行广播，因为广播通信属于二层通信。正因如此，才想到用VLAN来缩小广播域。试想一下，一个大型的交换网络，一旦有广播流量，就会向交换网络中的所有节点复制并发送一个广播包，这样就会给整个网络带来不小的负荷，可能会影响正常的数据交换。如果划分成了一个个VLAN，每个VLAN相当于一个小的独立二层交换网络，也就是一个小的广播域，这样每个VLAN中的广播包就只会在本VLAN中广播，影响范畴和程度自然就会大大降低。

6.10.2 理解VLAN的形成和工作原理

VLAN可按照功能、项目组、部门、IP子网、网络协议或者应用策略等方式对不同用户进行分组，而可以不考虑用户的实际物理位置。

VLAN具有与物理交换网络相同的属性，但是可以聚合即使不在同一个物理网段中的终端站点。在VLAN的配置与使用中，因为许多读者朋友并没有真正了解VLAN的形成原理，所以导致一些VLAN配置和VLAN路由、桥接出现故障时读者无法理解故障原因。

1.同一物理交换机中的VLAN

其实理解VLAN这个术语的关键就是要理解“虚拟”这两个字。虚拟表示VLAN所组成的是一个虚拟或者说是逻辑的LAN，并不是一个物理LAN。一个交换机中的各个VLAN可以理解为一个个虚拟交换机，如图6-47所示的物理交换机中就划分了五个VLAN，相当于有五个相互只有逻辑连接关系的虚拟交换机。

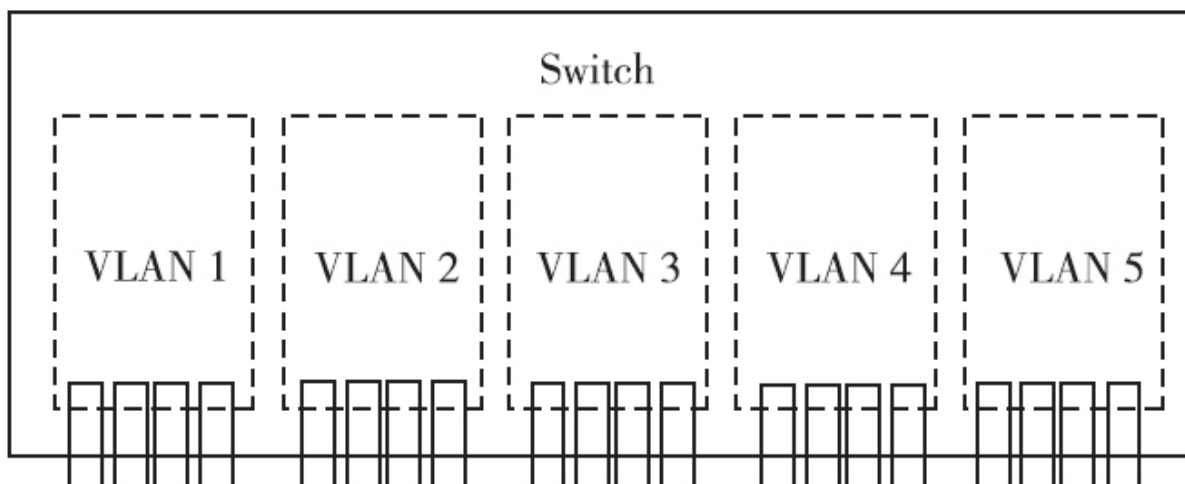


图 6-47 一台物理交换机中划分的多个VLAN

其实我们只要把一个VLAN看成一台交换机（只不过它是虚拟交换机），许多问题就比较好理解了，因为虚拟交换机与物理交换机具有相同的基本属性。同一物理交换机上的不同VLAN之间就像永远不可能有物理连接、只有逻辑连接的不同物理交换机一样。既然没有物理连接，那不同VLAN肯定是不能直接相互通信的（这里仅指二层通信），即使这些不同VLAN中的成员都处于同一IP网段。

位于同一VLAN中的端口成员就相当于同一物理交换机上的端口成员一样，不同情况仍都可以按照物理交换机来处理。如同一VLAN中的各成员可以属于同一网段，也可以属于不同网段，但通常是把属于同一网段的节点划分到同一VLAN中。如果都属于同一个网段，则肯定可以相互通信，就像同一物理交换机上连接同一网段的各个主机用户一样；但如果同一VLAN中的端口成员属于不同网段，则相当于一台物理

交换机上连接处于不同网段的主机用户一样，这时肯定得通过路由或者网关配置来实现相互通信。

2.不同物理交换机中的VLAN

因为一个VLAN中的端口成员不是依据成员的物理位置来划分的，所以通常是位于网络中的不同交换机上，也就是说一个VLAN可以跨越多台物理交换机，这就是VLAN的中继（Trunk）功能，如图6-48所示。这时我们就不要总按照物理交换机来看待用户的分布了，而是要从VLAN角度来看待了。如在图6-48所示中，就不要把它当成两台物理交换机，而是要把它当成是五台，且是两台物理交换机中相同的两个VLAN间有物理连接关系（就是两台物理交换机间的那个连接）的物理交换机。

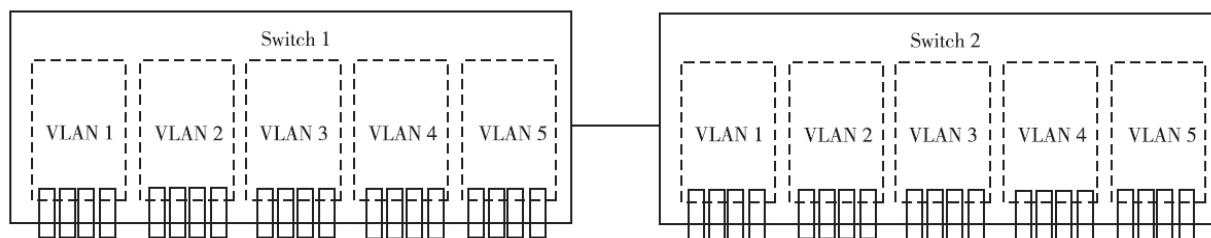


图 6-48 不同物理交换机上的相同VLAN

在不同交换机上可以有相同的VLAN，而且这些不同物理交换机上的相同VLAN间是相通的，是可以相互访问的，当然这得在物理交换机连接的端口允许这些VLAN数据包通过，这就是VLAN中的Trunk（中继）端口的功能了。

在同一物理交换机上的不可能存在两个相同的VLAN，但在不同交换机上可以存在多个相同的VLAN（其实是一台交换机上一个VLAN的延伸），而默认情况下只有相同VLAN中的成员才可以直接通信（不需要路由和桥接），所以在同一物理交换机上默认情况下不同VLAN间是不能直接通信的，即使它们都位于同一IP网段；但位于不同物理交换机上的相同VLAN却是可以直接进行二层通信的，只要物理交换机间的连接端口允许相应VLAN数据包通过学习即可，因为位于不同物理交换机上的相同VLAN间的连接就是物理交换机间的物理连接。

经验之谈 这里要区分VLAN中继和中继端口这两个概念。VLAN中继是指在一台交换机上的VLAN配置可以传播、复制到的网络中相连的其他交换机上，这就是VTP（VLAN中继协议）；而Trunk（中继）端口则是指在一个交换机端口允许一个或多个VLAN通信到达网络中相连的另一台交换机上相同的VLAN中。这是两个不同的概念。

3.VLAN间的互访

VLAN是二层协议，VLAN的虚拟或者逻辑属性决定了这些VLAN之间没有物理二层连接（只有逻辑连接），彼此独立，相当于一个个独立的二层交换网络。在不可能进行二层互访的情况下，我们只能通过三层来解决它们之间的连接问题。

一个独立交换网络与另一个独立交换网络进行三层连接有两种方式：一种是通过网关实现，另一种是通过路由实现。在不同VLAN间的逻辑连接也有这两种方式，其中每个VLAN的那个交换机虚拟接口

（SVI）就是对应VLAN成员的网关。为每SVI配置好IP地址，这个IP地址就是对应VLAN成员的网关IP地址。这种通过SVI进行的VLAN间成员互访的基本结构如图6-49所示。每个VLAN成员与其他VLAN成员进行通信时都必须通过双方作为各自VLAN成员网关的SVI。而每个VLAN内部的成员端口一般都是二层访问端口，直接连接PC用户。

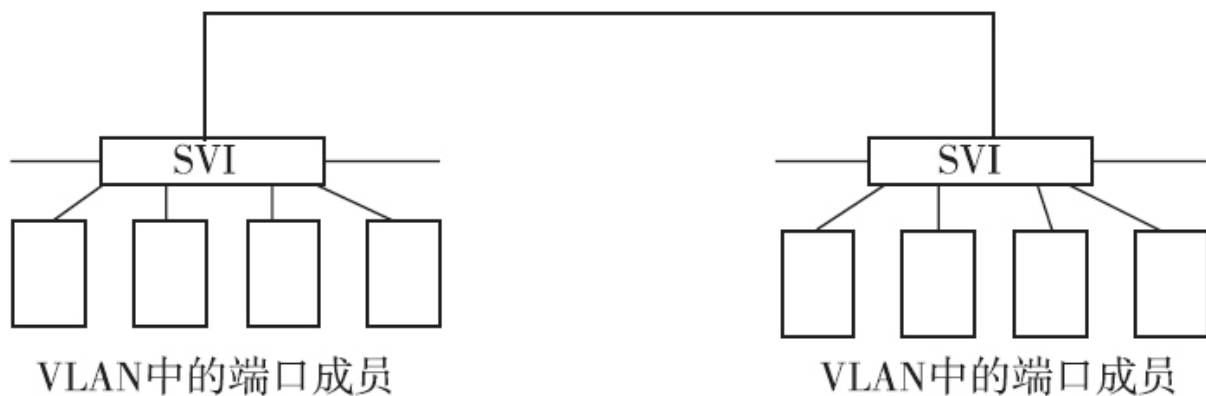


图 6-49 不同VLAN间通过SVI进行的逻辑连接示意图

通过路由方式来实现不同VLAN间的连接可以理解为在图6-49所示的两个SVI间加了一个提供路由功能的设备，可以是路由器（通过静态路由或各种路由协议实现），也可以是有三层交换模块的三层交换机（通过开启IP路由功能实现）。但各个VLAN对外还是以各自的SVI呈现的，各VLAN内部还是根据二层的MAC地址进行寻址的。当然这是在假设相同VLAN中的成员都是在同一网段的情况下。

如果同一个VLAN中的端口成员不在同一个IP网段，则需要像一台物理交换机上连接了多个网段的主机一样配置路由或网关来实现VLAN内部各成员的相互通信。如果是网关方式，则可以为该VLAN的SVI分配多个对应不同网段的IP地址，这些IP地址就相当于多个网关了，只要在对应网段的主机上配置指向SVI上的对应IP地址的网关，就可以与同一VLAN中其他网段的成员进行通信了。有关VLAN间的路由配置相对比较复杂，在此不做具体介绍了，感兴趣的读者可以参见笔者编著的《Cisco/H3C交换机配置与管理完全手册》（第2版）一书。

6.10.3 IEEE 802.1q帧头部格式

既然VLAN是一个二层协议，在交换机上启用了它后自然就会在帧上进行封装，添加VLAN协议头。支持VLAN的交换机的以太网帧就不再像图6-24所示的那样了，其还需要在以太网帧Data字段前中加上用于标记帧VLAN信息的4个字段（共4字节）IEEE 802.1q VLAN协议头，如图6-50所示。其中Priority、CFI和VLAN ID这三个字段统称为TCI（Tag Control Information，标签控制信息），占2字节。

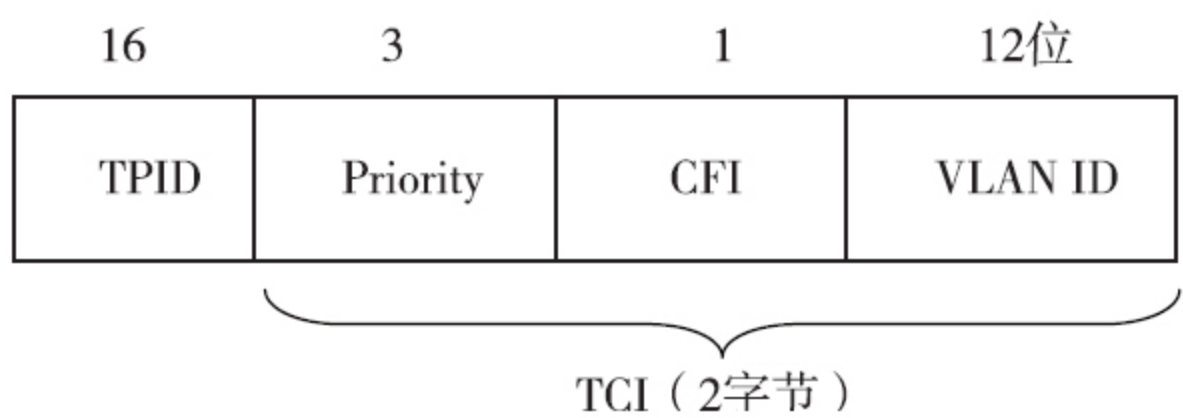


图 6-50 IEEE 802.1q协议头格式

说明 虽然有了VLAN协议，但要注意的是，并不是所有设备都支持VLAN协议，如我们的主机、打印机等终端设备，以及一些傻瓜式交换机都是不支持VLAN协议的，所以它们只能发送和接收如图6-20所示的不带VLAN标签的以太网帧。那么这个VLAN标签是在哪里加上去的呢？它是在支持VLAN协议的交换机上，连接主机和终端设

备的二层访问端口（Access Port）上加上去的。当把一个二层交换机端口添加到一个VLAN中后就具有了对应VLAN信息，当它接收到来自对应VLAN的主机发来的帧时，就会在帧中加上这个VLAN标签，以标记该帧是来自对应的VLAN。有关交换机的各种端口类型及收发数据规则，参见笔者编著的《Cisco/H3C交换机配置与管理完全手册》（第2版）一书。

图6-50中所示的4个字段的说明如下：

□TPID: Tag Protocol Identifier（标签协议标识符）字段，占2字节（16位），表明这是一个添加了IEEE 802.1q标签的帧（区别于未加VLAN标记的帧），值固定为0x8100（表示封装了IEEE 802.1q VLAN协议）。

□Priority: User Priority（用户优先级）字段，占3位，表示0~7八个优先级（数值越大，优先级越高），主要用于当交换机阻塞时，优先发送哪个数据帧，也就是QoS（服务质量）的应用，是在IEEE 802.1p规范中被详细定义的。

□CFI: Canonical Format Indicator（标准格式指示器）字段，占1位，用来兼容以太网和令牌环网。用来标识MAC地址在传输介质中是否以标准格式进行封装，取值为0表示MAC地址以标准格式进行封

装，为1表示以非标准格式封装，默认取值为0，在以太网中该值总为0，表示以标准格式封装MAC地址。

经验之谈 在这里要介绍一个绝大多数图书和文章都没有介绍的问题，那就是什么是标准格式MAC地址，什么是非标准格式MAC地址。其实，以太网（IEEE 802.3）和令牌总线网（IEEE 802.4）在传输介质中发送MAC地址字节是按从低到高的顺序（也就是我们平时写MAC地址格式中的从右到左的顺序）进行的，而令牌环网（IEEE 802.5）和IEEE 802.6标准中MAC地址字节在传输介绍中的发送顺序则相反，是从高到低的顺序。MAC地址字节的发送顺序也对应MAC地址字节的封装顺序，我们把IEEE 802.3和IEEE 802.4标准中的MAC地址封装顺序称为标准格式（canonical form），而把IEEE 802.5和IEEE 802.6标准中的MAC地址封装顺序称为非标准格式（non-canonical form）。例如一个MAC地址为12-34-56-78-9A-BC，以标准格式发送时，则它的比特次序为01001000 00101100 01101010 00011110 01011001 00111101，但是以非标准格式发送时，它的比特次序是00010010 00110100 01010110 01111000 10011010 10111100（注意，比较每个字节可以发现，它与前面的标准格式是次序相反的）。

□VLAN ID: VLAN IDentified（VLAN标识）字段，占12位，指明VLAN的ID，一共4096个，每个支持IEEE 802.1q协议的交换机发送出来的数据包都会包含这个域，以指明自己属于哪一个VLAN。

6.11 IEEE 802.1w协议

为了解决6.9.3节介绍的IEEE 802.1d STP协议的不足，在20世纪初IEEE推出了802.1w标准。它同样属于生成树协议类型，称为快速生成树协议（Rapid Spanning Tree Protocol，RSTP），作为对802.1D标准的补充。

那么为什么在有了IEEE 802.1d协议后，还要制定IEEE 802.1w协议呢？原来，IEEE 802.1d协议虽然解决了链路闭合引起的死循环问题，但是生成树的收敛（指重新设定网络中的交换机端口状态）过程仍需比较长的时间（30~50s）。于是IEEE 802.1w协议问世了，它使得收敛过程由原来的30~50s减少为现在的1~10s，因此IEEE 802.1w又称快速生成树协议（RSTP）。对于现在的网络来说，这个速度足够快了。

RSTP通过快速生成树算法在交换网络中阻断部分冗余路径，建立起无环路的树状网络。RSTP所采用的快速生成树算法与生成树算法一样，也是一个分布式算法。RSTP运行在一个桥接网络中的所有交换机（或者交换机）上，负责为该桥接网络计算出简单连通的树形活跃拓扑。计算时也是先选择一个交换机作为树根（即“根交换机”），同时为所有交换机的所有端口指定角色。

RSTP算法基本和IEEE 802.1D标准中定义的STP算法基本一样，唯一不同的是RSTP解决了STP算法对任何端口只要从Blocking（阻塞）状态转换到Forwarding（转发）状态必须经过2倍转发延时（包括由侦听状态到学习状态的等待时间和由学习状态到转发状态的等待时间）这个不足。它利用点对点连接，针对各种端口在拓扑中角色的不同，对某些端口实现了从阻塞状态直接转换到转发状态来提供快速的生成树收敛，可以在少于1秒的时间内重新配置生成树（在IEEE 802.1d的STP生成树中默认是50s）。

1.RSTP的主要改进

RSTP在STP基础上做了以下3个重要改进（也是RSTP的主要优势），使得收敛速度快得多（最快1s以内）：

1) 为根端口和指定端口设置了快速切换用的替换端口（Alternate Port）和备份端口（Backup Port）两种角色。当根端口或指定端口失效时，替代端口或备份端口就会无延时地进入转发状态。

图6-51中所有交换机都运行RSTP协议，交换机1是根交换机，图6-51a中设定交换机3的2号端口为端口1的备份端口，备份端口1中的路径信息，这样在端口1的路径信息丢失后，就可以由端口2中得到，故端口2成为端口1的替代端口。替代端口的作用是用在主端口出现故障时，快速接替主端口的工作。图6-51b假设交换机3的端口1是根端口，

交换机2上的端口2设置为交换机3的端口1的替代端口，同时进入阻塞状态。但当交换机3的端口1所在链路失效的情况下，交换机2上的端口2就能够立即进入转发状态，且无须等待2倍转发延时时间。

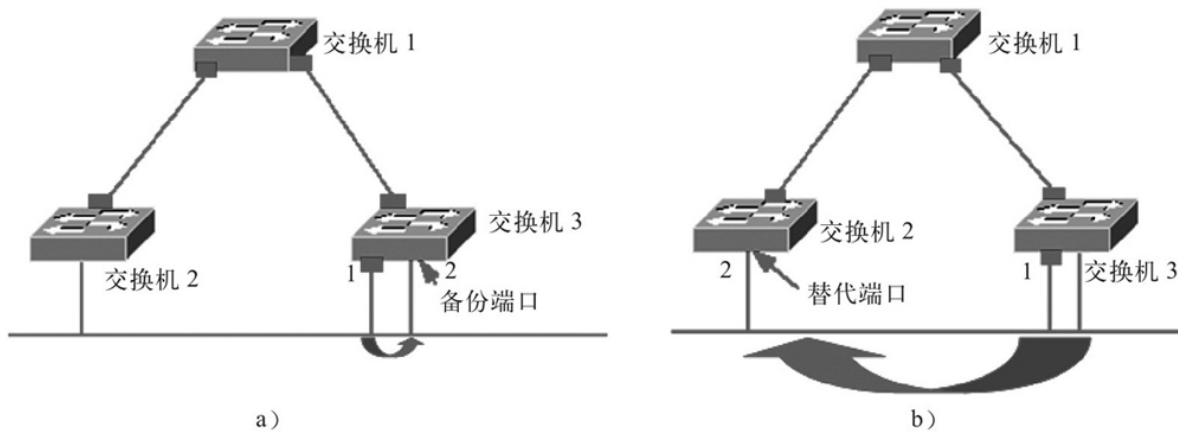


图 6-51 RSTP替代端口的作用

2) 在只连接了两个交换端口的点对点链路中，指定端口只需与下级交换机进行一次握手就可以无延时地进入转发状态。但如果是连接了3个以上交换机的共享链路，下级交换机是不会响应上级指定端口发出的握手请求的，其只能像STP一样等待2倍转发延时时间才能进入转发状态。

3) 把直接与终端（如PC机）相连而不是与其他交换机相连的端口定义为边缘端口（Edge Port）。边缘端口可以直接进入转发状态，不需要任何延时。但由于交换机无法知道端口是否是直接与终端相连，所以需要人工配置。

由上可见，**RSTP**协议相对于**STP**协议的改进主要体现在端口状态上。为了支持这些改进，**BPDU**的格式做了一些修改，但**RSTP**协议仍然向下兼容**STP**协议，可以混合组网。

2.RSTP的不足

虽然如此，**RSTP**和**STP**一样同属于单生成树**SST**（Single Spanning-Tree），所以它仍具有以下不足：

- 1) 由于整个交换网络（其实就是一个**VLAN**网络）只有一棵生成树，在网络规模比较大的时候会导致较长的收敛时间，拓扑改变的影响面也较大。
- 2) 在网络结构不对称的时候，单生成树就会影响网络的连通性。
- 3) 当链路被阻塞后将不承载任何流量，造成了带宽的极大浪费，这在环型城域网的情况下比较明显。

这些缺陷都是单生成树**SST**无法克服的，于是支持**VLAN**的多生成树（**MST**）协议出现了，这就是下节要介绍的**IEEE 802.1s**协议。

6.12 IEEE 802.1s协议

前面介绍的IEEE 802.1d STP和IEEE 802.1w RSTP都是针对单一生成树实例（把整个交换网络看成一个生成树）进行应用的。Cisco的PVST、PVST+和Rapid-PVST+尽管属于多生成树实例，但其是基于VLAN的，每个实例对应一个VLAN，这样一来不仅生成树实例可能会非常多，难以管理，还没有一个容错机制，容易出现单点失效。这么多生成树实例维护起来比较困难，而且为每个VLAN每隔2s就发送一个BPDU，交换机也是难以承受的。

为了解决这一问题，IEEE工作组又开发了新的生成树协议--MSTP（Multiple Spanning Tree Protocol，多生成树协议），对应的标准就是IEEE 802.1s。MSTP可以对网络中众多的VLAN进行分组，一些VLAN分到一个组里，另外一些VLAN分到另外一个组里。这里的组就是后面讲的MST（多生成树实例）。每个实例对应一个生成树，BPDU是只在实例内部进行发送，这样所发送的BPDU数量明显减少了，减少了交换机的通信负担。

说明 在IEEE 802.1s MSTP发布以前，Cisco使用的是自己开发的MISTP（Multiple Instance Spanning Tree Protocol，多实例生成树协议）。在IEEE正式发布MSTP后，Cisco就直接采用IEEE发布的MSTP协议，原来的MISTP就成了预标准。

6.12.1 MSTP简介

MSTP是IEEE 802.1s标准中定义的一种新型多实例化生成树协议。它提供了快速收敛和在一个VLAN环境下实现负载均衡双重优势。MSTP比PVST（Per-VLAN Spanning Tree，每VLAN生成树）、PVST+（per-VLAN Spanning Tree Plus，增强型每VLAN生成树）收敛更快，并且与STP、RSTP和PVST+生成树架构兼容。有关PVST和PVST+生成树协议可参见笔者编著的《Cisco/H3C交换机配置与管理完全手册》（第2版）一书。

MSTP允许通过VLAN中继来构建多个生成树，可以组合和关联多个VLAN到生成树实例（Spanning Tree Instance，SPI）。每个实例可以有一个独立于其他生成树实例的拓扑。这种新的架构为数据通信和负载均衡提供了多个转发路径，也提供了网络容错机制，因为一个实例（也就是一个转发路径）的失效不会影响其他实例。

与PVST+中所有生成树实例都是独立的不一样，MSTP建立、管理两种生成树类型：

- 1) IST（Internal Spanning Tree，内部生成树）：是在MST区域中运行的生成树总称。

IST是MST区域中的一个特殊生成树实例（也可以MST0表示），在一定程度上代表了一个MST区域，因为它在一个MST内部的所有链路上都是活跃的，专为其他MST实例提供拓扑信息服务。在一个MST区域中，IST是不能被删除的，也是自动存在的，但还可以手动创建其他MST实例，这些其他MST实例号只能在1~4094之间，通常是以MSTn（n=1~4094）进行标识。默认情况下，所有VLAN是分配到IST实例中的，但在实际配置中，往往不把任何VLAN分配到这个实例中，因为IST只用于生成树BPDU的收发，不用于数据通信。

IST仅发送和接收BPDU的生成树实例，所有其他生成树实例信息包含在它的MST记录（MSTP Record，又称M记录）中，然后用MSTP BPDU进行封装。因为MSTP BPDU携带了所有实例信息，这样在支持多个生成树实例时所需要处理的BPDU数量就会大大减少。

在同一个MST区域中的所有MST实例共享相同的协议计时器，但是每个MST实例有它们自己的拓扑参数，如根网桥ID、根路径开销等。默认情况下，所有VLAN都是指派到IST实例中。但是，一个MST实例与所在区域相关。例如，区域A中的MST实例1与区域B中的MST实例1是无关的，即使区域A和区域B是互连的。

2) CST（Common Spanning Tree，公共生成树）是用来互联不同MST区域的单生成树实例。如果把每个MST区域看做一个设备，CST就是这些设备通过STP协议、RSTP协议计算生成的一棵生成树。在每

个MST区域中计算出的生成树都是作为包含整个交换域的CST的子树出现的。接收到来自其他MST区域BPDU的交换机则被称为边界

（Boundary）交换机，对应的链路称为边界链路，对应的端口就称为边界端口，如图6-52所示。

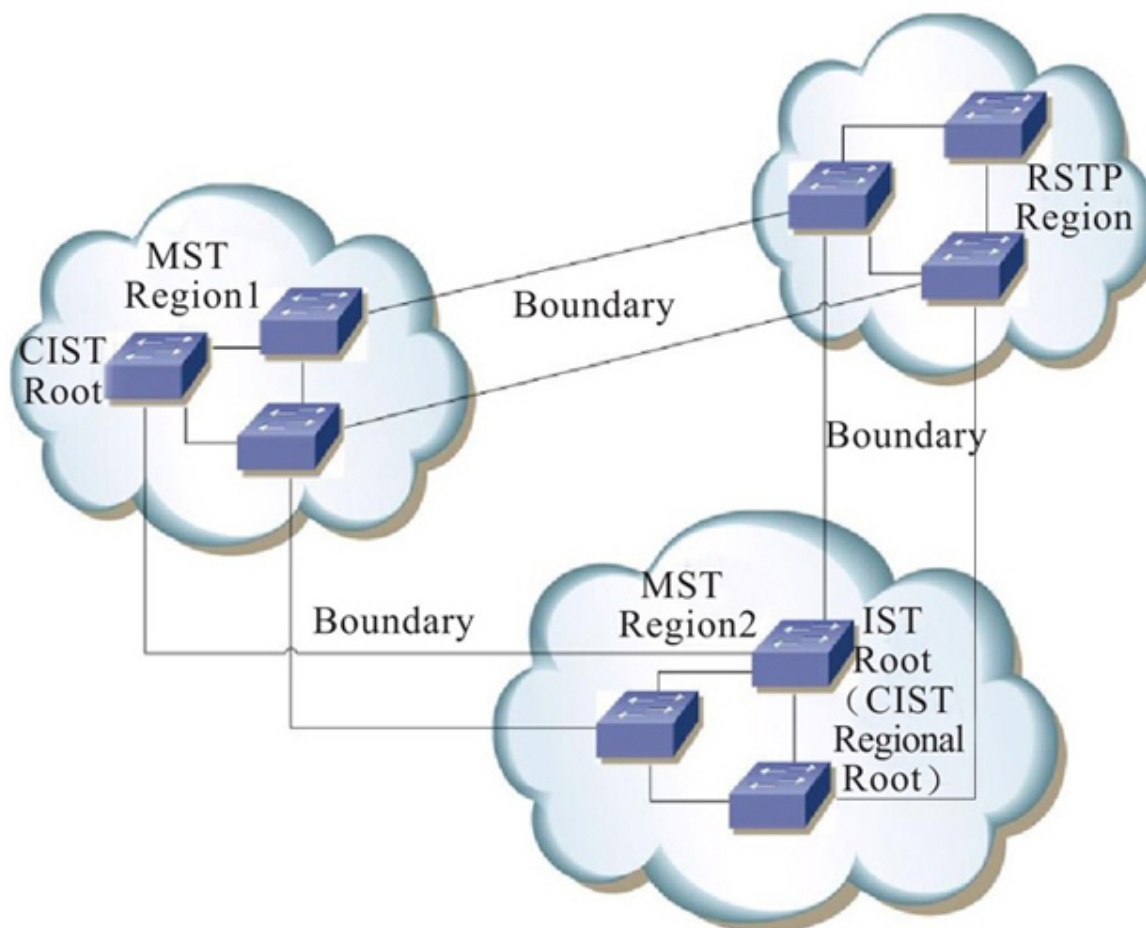


图 6-52 MST区域上的互联

3) CIST（Common and Internal Spanning Tree，公共和内部生成树）：是一个MST区域中所有IST、连接MST区域的CST（Common

Spanning Tree，公共生成树）和其他SST（Single Spanning Trees，单生成树）的集合。

由上可知，从作用的范围来看IST是最小的，仅属于一个MST区域内部，CST次之，是MST区域间的互联生成树实例，而CIST最大，包括了IST和CST。它们的作用范围关系可以用图6-53所示来描述。但要知道的是，每个MST区域中的各实例生成树都是作为CST的子树。

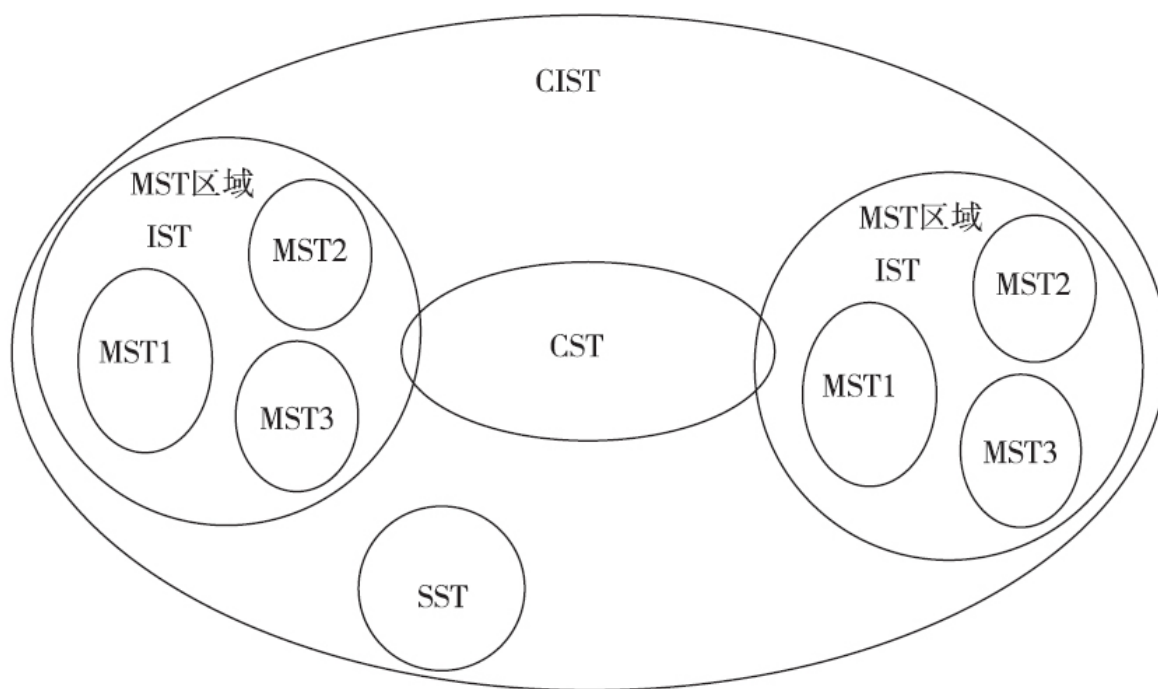


图 6-53 IST、MST_n、SST、CST和CIST之间的作用范围关系

除了以上两种生成树类型外，在理解MSTP时还涉及几个与IST、CIST相关的术语。

1) CIST根 (CIST root) : 是跨越整个网络的唯一实例——CIST实例的根网桥。除了发送IST配置BPDU外, 每个交换机初始化时都宣告自己作为CIST根。交换机连同IST配置BPDU一起传递CIST配置信息。在一个MST区域内部的交换机永远不会改变到达CIST根的路径开销, 详见下面要说到的CIST外部根路径开销。

在边界端口上 (如图6-52所示), 交换机仅交换它们的CIST BPDU。也就是说, 在边界上的交换机是隐藏MST区域中的IST信息的, 但是会传递CIST度量。在边界链路上通常会发生RSTP同步进程 (因为启用MSTP时是会同时启用RSTP的), 最终在所有区域中, 具有最低BID (网桥ID=优先级值+MAC地址) 的交换机被选举为CIST根。但要注意, 每个区域还会选举一个本地IST根的作为CIST区域根。这在下面将会说到。

2) CIST外部根路径开销 (CIST External Root Path Cost) : 是到达CIST根的开销。这个开销在一个MST区域中是保持不变的, 因为MST区域对于CIST来说就像一台单一虚拟交换机。外部根路径开销仅计算边界链路上的开销, 而不计算区域内部的开销。本质上, CIST外部根路径开销信息是以隧道的方式穿越MST区域的。

3) CIST区域根 (CIST Regional Root) : 在Cisco MSTP预标准中被称为IST主 (IST Master)。如果CST根在区域中, CIST区域根就是

CIST根；否则，CIST区域根就是到达CIST根最近的交换机。CIST区域根是作为IST的根交换机。

包含CIST根的区域，同时也会宣告该交换机作为本地IST根。但是对于不包含CIST根的区域就不是这样了。在不包含CIST根的区域中，在所有连接其他MST区域的边界交换机中选举一个作为IST根（也就是CIST区域根）。这个选举过程是基于最低的CIST外部根路径开销的，不同于前面所说的纯粹依据BID的CIST根的选举，这种选举是在一个MST区域内部进行。但是，如果出现有两台交换机具有相同的CIST根外部路径开销时，就会再依据交换机的BID进行再次选举了。MSTP会阻塞所有标记为到达CIST根的替代（alternate）路径的冗余边界uplinks（上行链路）。

在一个MST区域内，交换机建立常规IST，使用CIST区域根作为IST根。但是要注意，这棵生成树使用存储在本地ISL BPDU中的内部根路径开销。这种开销会按照区域内部所有链路进行递增，但是它永远不会超出区域范围。在区域之间，交换机之间仅交换CIST外部根路径开销信息。

4) CIST内部根路径开销（CIST Internal Root Path Cost）：是一个MST区域中交换机到达CIST区域根的开销。这个开销仅与IST（实例0）有关。

6.12.2 MST区域及工作原理

对于参与到MST实例的交换机，必须考虑为这些交换机配置相同的MST配置信息。具有相同MST配置信息的互联交换机就组成了一个MST区域。图6-54所示的就是一个MST区域划分示例。

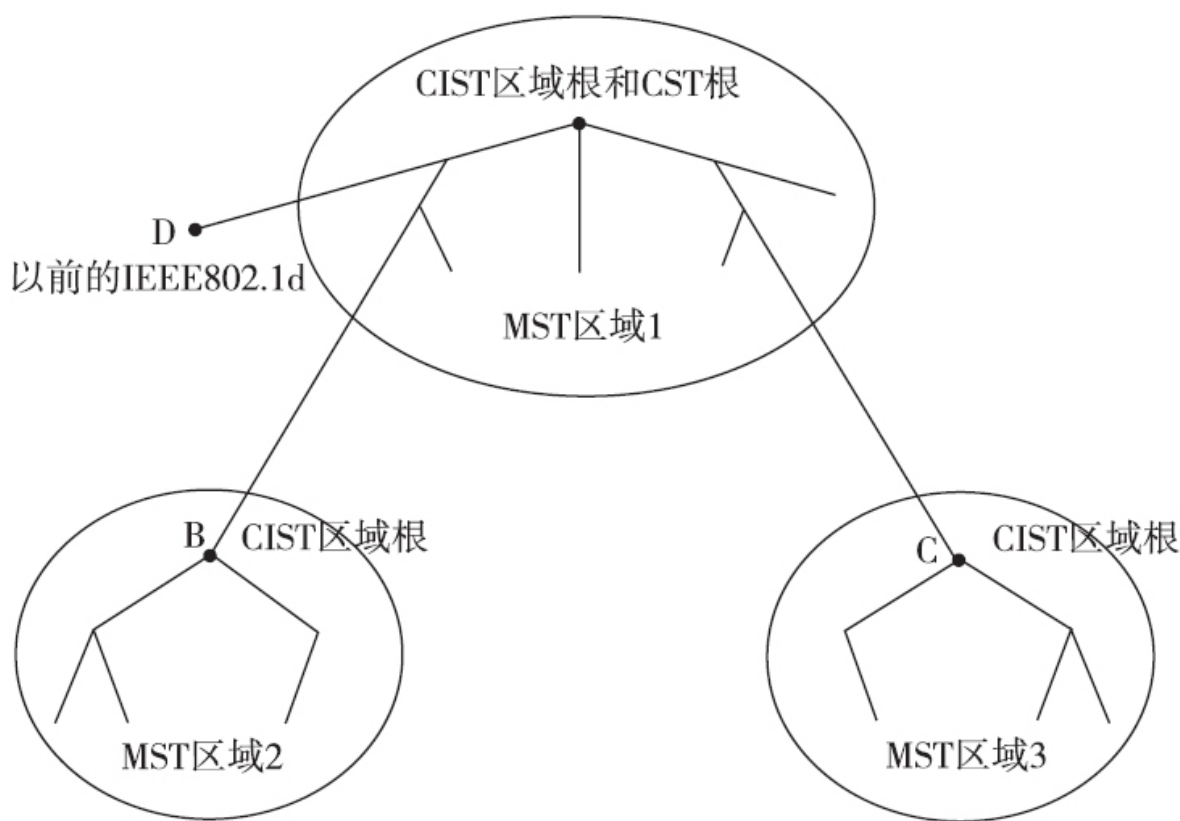


图 6-54 MST区域示例

MST配置（MST Configure）控制每个交换机属于哪个MST区域。其配置包括区域名称、版本号和MST VLAN到实例的分配映射。你可以使用spanning-tree mst configuration全局配置模式命令来配置区域中的

交换机，这样就进入了MST配置模式。在这个模式下，你可以通过instance MST配置模式命令映射多个VLAN到一个MST实例中，通过name MST配置模式命令指定MST区域名，通过revision MST配置模式命令设置修订号。

一个区域可有一个或多个具有相同MST配置的成员，每个成员都必须具有处理RSTP BPDU的能力。一个网络的MST区域中的成员数是没有限制的，但是每个区域最多只支持65个生成树实例（是在IOS 12.25XH及以上版本时），也就是说最多有65个VLAN组。实例可以由0~4094范围中的任一数字标识。在同一时刻，仅可以把一个VLAN分配到一个生成树实例。

1.在MST区域内部的工作原理

IST生成树与MST区域中所有运行MSTP的交换机连接。在IST收敛时，IST的根交换机将成为CIST区域根（在IEEE 802.1s标准以前称之为IST主）。它是区域中具有最低网桥ID，到达CIST根网桥路径开销最小的交换机。注意，CIST区域根与CIST根是不一样的，但如果在网络中仅有一个区域，则CIST区域根也就是CIST根。如果CIST根位于区域之外，则位于区域边界的MSTP交换机将被选举为CIST区域根。

当一个MSTP初始化时，该MSTP协议将发送BPDU，要求把自己当成CIST根和CIST区域根，并且把到达CIST根和到达CIST区域根的路

径开销都设为0。运行MSTP协议的交换机也会初始化它的所有MST实例，把自己当成这些MST实例的根交换机。如果这台交换机接收到一个比自己当前更高级的MST根信息（更低的网桥ID、更低的路径开销等），则它会放弃把自己当成CIST区域根。

在初始化过程中，一个区域下可以有多个子区域（因为一开始每个交换机都把它自己当成区域根），每个子区域都有它自己的CIST区域根。若交换机接收到更高级的IST信息，则这些交换机会离开原来的子区域，而加入到包含正确CIST区域根的新子区域中。这样，子区域数会减少，除了包含正确CIST区域根的那个子区域。

正确情况下，在同一个MST区域中的所有交换机必须接受同一个CIST区域根，所以，在区域中的任何两台交换机如果要收敛到同一个CIST区域根，则仅需要同步其在一个MST实例中的端口角色即可。

2.MST区域间的工作原理

如果网络中有多个区域，或者有运行IEEE 802.1d STP的交换机，MSTP需要建立并维护CST，它包括网络中所有的MST区域和所有的STP交换机。MST实例与区域边界的IST实例一起形成CST。

IST连接MST区域中的所有MSTP交换机，在整个交换域的CIST中，IST是以一个子生成树呈现的。这个子生成树的根就是CIST区域

根。对于邻接的STP交换机和MST区域来说，本地交换机上的MST区域表现为一个虚拟交换机。

在图6-54中包括了多个MST区域和一个运行IEEE 802.1d STP的交换机（交换机D）。区域1的CIST区域根也作为CIST根，区域2中的CIST区域根和区域3中的CIST区域根分别是各自CIST区域中的子生成树的根。

在MST区域间，仅CST实例发送和接收BPDU，MST实例添加他们的生成树信息到与邻接交换机交互的BPDU中，并计算最终的生成树拓扑。正因如此，与BPDU相关的参数（如Hello时间、转发时间、最大生存时间和最大跳数等）在CST实例上配置，这个过程会影响所有MST实例。与生成树拓扑相关的参数（如交换机优先级、端口VLAN开销和端口VLAN优先级）可以同时CST实例和MST实例上配置。

MSTP交换机使用v3版本的RSTP BPDU或者IEEE 802.1d STP BPDU与邻接的IEEE 802.1D交换机通信。在MSTP交换机之间使用MSTP BPDU进行通信。

6.13 IEEE 802.1x协议

IEEE 802.1x标准定义了一个C/S模式的访问控制和认证协议，以阻止未授权客户端通过公共可访问端口连接LAN。认证服务器在交换机或者LAN提供可用服务之前为每个连接在交换机端口上的客户端提供认证。在通过认证以前，IEEE 802.1x访问控制仅允许EAPOL

（Extensible Authentication Protocol over LAN，基于局域的可扩展认证协议）、CDP（Cisco Discovery Protocol，思科发现协议）、STP

（Spanning Tree Protocol，生成树协议）这类控制通信通过端口到达所连接的客户端；在认证成功后，其他数据或语音通信就都可以在端口上通过了。

6.13.1 IEEE 802.1x认证设备角色

在整个IEEE 802.1x基于端口认证体系网络中，包括了三种基本角色的设备，如图6-55所示。

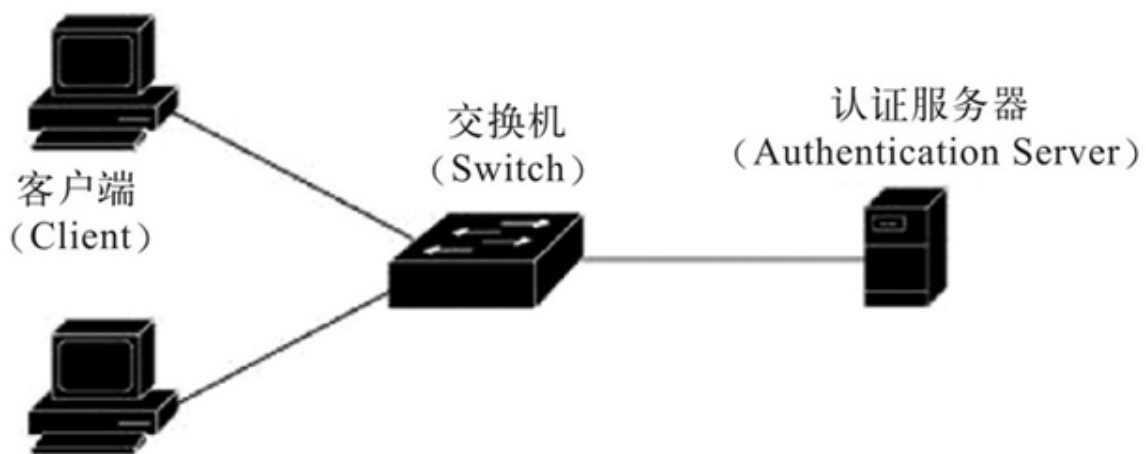


图 6-55 IEEE 802.1x基于端口认证设备角色

□客户端（Client）：是指连接交换机端口的工作站主机，是请求访问交换机和LAN服务，并从交换机上得到响应的工作站设备。这个工作站设备必须运行与IEEE 802.1x标准兼容的客户端软件，如Windows XP/7等系统。这里的客户端角色在IEEE 802.1x标准中是称为恳求者（Supplicant）。

□认证服务器（Authentication server）：执行实际的客户端认证工作的主机设备，通常是运行RADIUS服务器程序的主机。认证服务器确认客户端身份的合法性，并通知交换机对应客户端是否通过了认证，是否可以访问局域网和交换机服务。因为交换机担当中间代理角色，所以认证服务对于客户端来说是透明的。目前在Cisco IOS交换机中仅支持RADIUS服务器作为IEEE 802.1x认证服务器。

□交换机（Switch，可以是边缘交换机或无线AP）：对客户端提供基于客户端认证状态的网络访问控制。交换机在客户端和认证服务器之间起一个中间代理角色，向客户端请求标识信息，通过认证服务验证客户端标识信息，同时向客户端中继转发认证服务器对客户端的请求响应。交换机包括RADIUS客户端软件，负责EAP帧封装和解封装。这里的交换机角色在IEEE 802.1x标准中是称为认证者（Authenticator）。

在交换机接收到来自客户端的EAPOL帧时，中继转发到认证服务器，然后去掉以太网头部，保留EAP帧，并以RADIUS帧格式重新封装。EAP帧在封装过程中不做修改，RADIUS认证服务器必须在本地（native）帧格式中支持EAP。当交换机收到来自认证服务器的EAPOL帧时，也会去掉认证服务器的头部，同时也只保留EAP帧，然后重新加上以太网头进行再封装，然后发送给客户端。

6.13.2 IEEE 802.1x主机模式

IEEE 802.1x端口的主机模式决定了在该端口上连接了多个客户端时是否允许多个主机被认证，以及如何强制认证。配置IEEE 802.1x端口可以使用以下五种模式中的任意一种，另外，其他每种模式可以修改为允许预认证开放访问模式：

- ☐单主机模式（Single-Host Mode）；
- ☐多主机模式（Multiple-Hosts Mode）；
- ☐多域认证模式（Multidomain Authentication Mode）；
- ☐多认证模式（Multiauthentication Mode）；
- ☐预认证开放访问（Preauthentication Open Access）。

1.单主机模式

可以配置IEEE 802.1x端口为单主机模式或多主机模式（Single-Host Mode）。在单主机模式中，通常是一个交换机端口下连接一台客户端，而不是连接一个网段（如图6-56所示），且仅有一个客户端可以连接在已启用IEEE 802.1x的端口上。

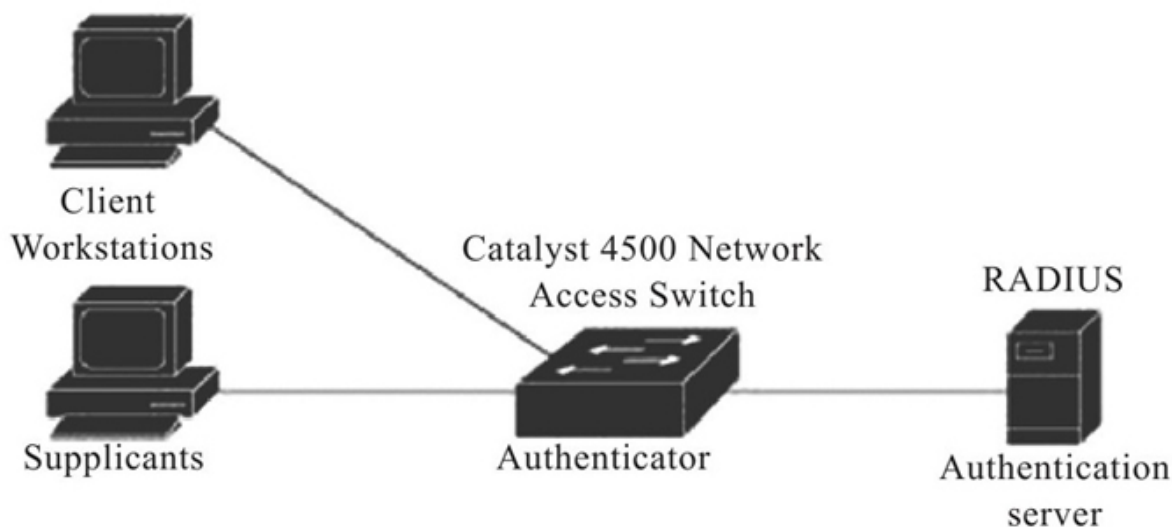


图 6-56 单主机模式IEEE 802.1x认证示例

在这种模式中，交换机从链路状态改变至打开状态后，通过发送一个EAPOL帧来检测客户端。如果客户端离开，或者接上其他客户端，交换机会改变端口链路状态为关闭，端口返回为未授权状态，不允许新的客户端访问网络。这就是单主机的意思。

2.多主机模式

在多主机模式（Multiple-Hosts Mode）下，可以在一个启用了IEEE 802.1x的端口上连接多个主机，如一个交换机端口下面连接了另一台交换机，或者WLAN AP。图6-57所示为在WLAN中IEEE 802.1x基于端口的认证。在这种模式中，AP上所有连接的客户端只要其中一个经过了授权，则所有客户端都可以进行网络访问。如果这个端口没有经过授权，则交换机拒绝所有连接客户端的访问。在这种拓扑中，WLAN

AP负责为它所连接的客户端进行认证，同时它又作为交换机的客户端。

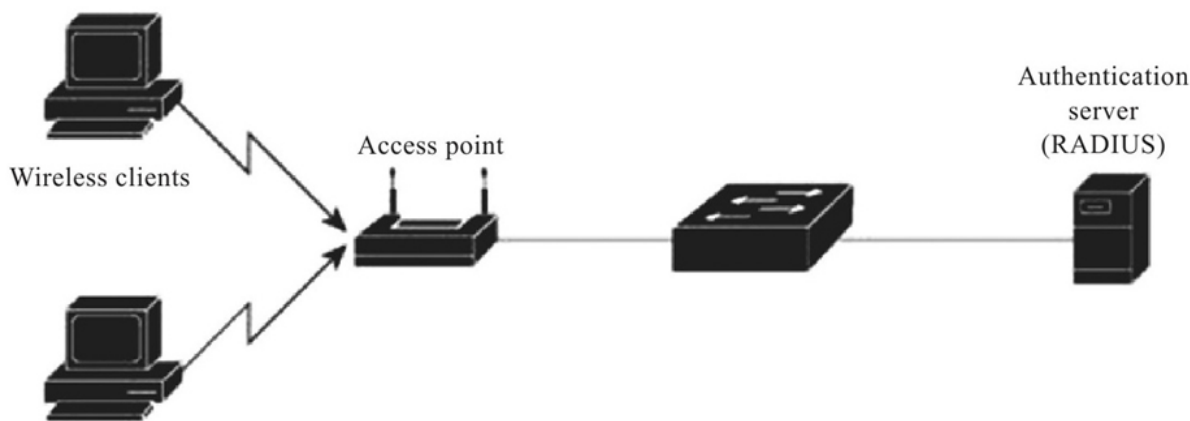


图 6-57 多主机模式IEEE 802.1x认证示例

在启用了多主机模式后，可以使用IEEE 802.1x认证功能来验证端口，使用端口安全功能管理所有MAC地址的网络访问，包括客户端的MAC地址。

3.多域认证模式

多域认证（Multidomain Authentication，MDA）允许IP电话和位于IP电话后面的一个主机单独使用IEEE 802.1x、旁路MAC地址认证

（MAC authentication bypass，MAB）或者基于Web的认证（仅限主机）。多域是指数据和语音两个域，每个端口仅允许两个MAC地址的设备访问。交换机可以把主机放进数据VLAN中，而把IP电话放进语音VLAN中，即使它们连接在同一个交换机端口。数据VLAN和语音

LVAN可以在CLI中独立配置。设备标识为数据或者语音设备依据的是从AAA服务器上接收到的厂商特殊属性（vendor-specific-attributes, VSAs）。数据VLAN和语音VLAN也可以在认证过程中从AAA服务器接收的VSAs中获得。

图6-58所示为一个在启用了IEEE 802.1x端口上连接的IP电话后面再连接了单一主机的典型MDA应用。因为客户端不是直接连接交换机的，所以在客户端断开连接时交换机也不能检测到端口链路的丢失。为了阻止其他设备使用原客户端已建立的认证属性非法连接在这个已启用IEEE 802.1x端口上，Cisco IP电话会发送一个CDP主机已存在的类型-长度-值（TLV）包通知交换机连接客户端的端口链路状态已发生改变。

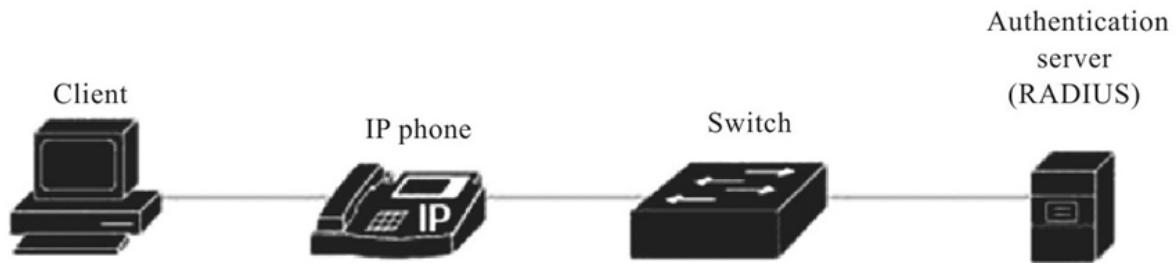


图 6-58 多域认证模式IEEE 802.1x认证示例

4.多认证模式

多认证模式（Multiauthentication Mode）允许一个语音VLAN中的客户端和多个数据VLAN中的客户端得到认证。当一个集线器或者AP

连接到一个启用了IEEE 802.1x端口时，多认证模式通过为每个客户端分别请求认证，为多个主机提供安全认证。对于不兼容IEEE 802.1x的设备，可以使用MAB（旁路MAC地址认证）或者基于Web认证方式作为个别主机的认证方法，允许在一个端口上为不同主机提供不同的认证方法。

多认证模式通过分配请求认证的设备到数据或者语音VLAN中来支持语音VLAN中的MDA（多域认证）功能。当一个端口处于多认证模式时，来宾VLAN和认证失败的VLAN将不再在数据设备上激活。

5.预认证开放访问

以上四种认证模式中的任何一种都可以通过额外配置来允许一个设备在通过认证前获取网络访问权限。这种预认证开放访问

（preauthentication open access）功能在像预启动执行环境（Pre-boot eXecution Environment，PXE，如远程启动设备）应用中非常有用，因为这类设备必须在启动时访问网络，从网络上下载包括认证客户端的可启动映像等文件。

在配置了主机模式后可通过authentication open命令启用预认证开放访问功能，作为配置的主机认证模式扩展。例如，如果在单主机模式下启用预认证开放访问功能，则端口上仅允许一个MAC地址访问。当启用预认证开放功能时，端口上的初始化通信仅受在端口上配置的其

他不依赖于IEEE 802.1x的访问控制方法限制。如果在端口上没有配置其他访问控制方法，则客户端设备可以全面访问所配置的VLAN。

6.13.3 IEEE 802.1x认证流程

交换机端口状态决定了所连接的客户端是否被允许访问网络。端口最初的状态是未授权状态，此时端口除了IEEE 802.1x协议包外不能接收和发送任何包。在端口通过IEEE 802.1x认证后，则该端口转换成授权状态，允许所有通信通过。

如果一个不兼容IEEE 802.1x的客户端连接到一个未授权的交换机端口上，则交换机会向该客户发送请求标识。但是，在此时该客户不会响应交换机的这个请求，继续保持未授权状态，该客户端也不允许访问网络。如果在该端口上配置了一个来宾VLAN，则该端口以未授权状态被置于这个来宾VLAN中。

相反，如果一个支持IEEE 802.1x的客户端连接到一个没有运行IEEE 802.1x协议的交换机端口中上，客户端通过发送EAPOL-start帧初始化认证过程。但是，此时，客户端不会收到交换机的响应，在多次发送请求仍没收到交换机的响应后，客户端就认为它所连接的端口是处于授权状态的，正常发送数据。

1.端口状态

在交换机设备中可以配置以下端口状态：

☐force-authorized: 强制授权，禁止IEEE 802.1x认证，使端口处于不需要IEEE 802.1x认证消息交换的授权状态。该端口无须经过IEEE 802.1x认证就可正常接收和发送客户端的通信。这是默认设置。

☐force-unauthorized: 强制非授权，使端口处于未授权状态，忽略客户端的所有认证请求。交换机不能为连接在这些端口上的客户端提供认证服务。

☐auto: 允许进行IEEE 802.1x认证，使端口最开始为未授权状态，仅允许接收和发送EAPOL帧。当对应端口所在链路状态由关闭到打开时，或者当接收到一个EAPOL-start帧时开始IEEE 802.1x认证。交换机向客户端请求标识，并开始在客户端和认证服务器之间中继认证消息。交换机可以使用客户端的MAC地址来唯一标识连接网络的每个客户端。

如果客户端成功通过认证，则其所连接的端口改变为授权状态，来自该客户端的所有通信均可通过此端口；如果认证失败，则此端口仍保持在未授权状态，但是仍将尝试认证。如果认证服务器不可达，则交换机可以重传客户端的认证请求；如果在指定次数重传后仍不能接收到来自认证服务器的响应，则认证失败，对应的客户端不允许访问网络。

如果对应端口的链路状态由打开变为关闭，或者端口上接收到一个帧，则此端口恢复为未授权状态。

2.IEEE 802.1x认证流程

若启用IEEE 802.1x基于端口认证，并且客户端支持IEEE 802.1x客户端，则将发生以下认证事件：

□如果客户端标识是有效的，并且成功通过IEEE 802.1x认证，则交换机允许客户端访问网络。

□如果在交换机等待来自客户端或者认证服务器的EAPOL消息交换过程中超时，并且启用了MAC地址认证旁路特性（MAC authentication bypass，也就是不进行IEEE 802.1x认证，直接采用MAC地址认证，这对一些不能安装IEEE 802.1x兼容软件的设备的认证非常必要，如打印机和特殊的终端），交换机可以使用客户端MAC进行认证。如果客户端MAC地址是有效的，且通过了MAC地址认证，则交换机允许客户端访问网络；如果客户端MAC地址是无效的，MAC地址认证失败，则交换机分配这个客户端到来宾VLAN（guest VLAN，在已配置了来宾VLAN的前提下），仅提供有限的服务。

□如果交换机从兼容IEEE 802.1x的客户端获得一个无效的客户端身份标识，并且指定了一个受限的VLAN（如来宾VLAN），则交换机可以分配该客户端到受限VLAN中，仅提供有限服务。

□如果RADIUS认证服务器不可用（关闭了），但启用了临界认证（critical authentication，即inaccessible authentication bypass，或者称为AAA失效策略）特性，则交换机通过把端口置于RADIUS服务器上配置的或者用户指定的访问VLAN的临界认证状态来允许客户端访问网络。

图6-59显示了IEEE 802.1x基于端口认证流程，当发生以下情形时，交换机会重新认证客户端的。

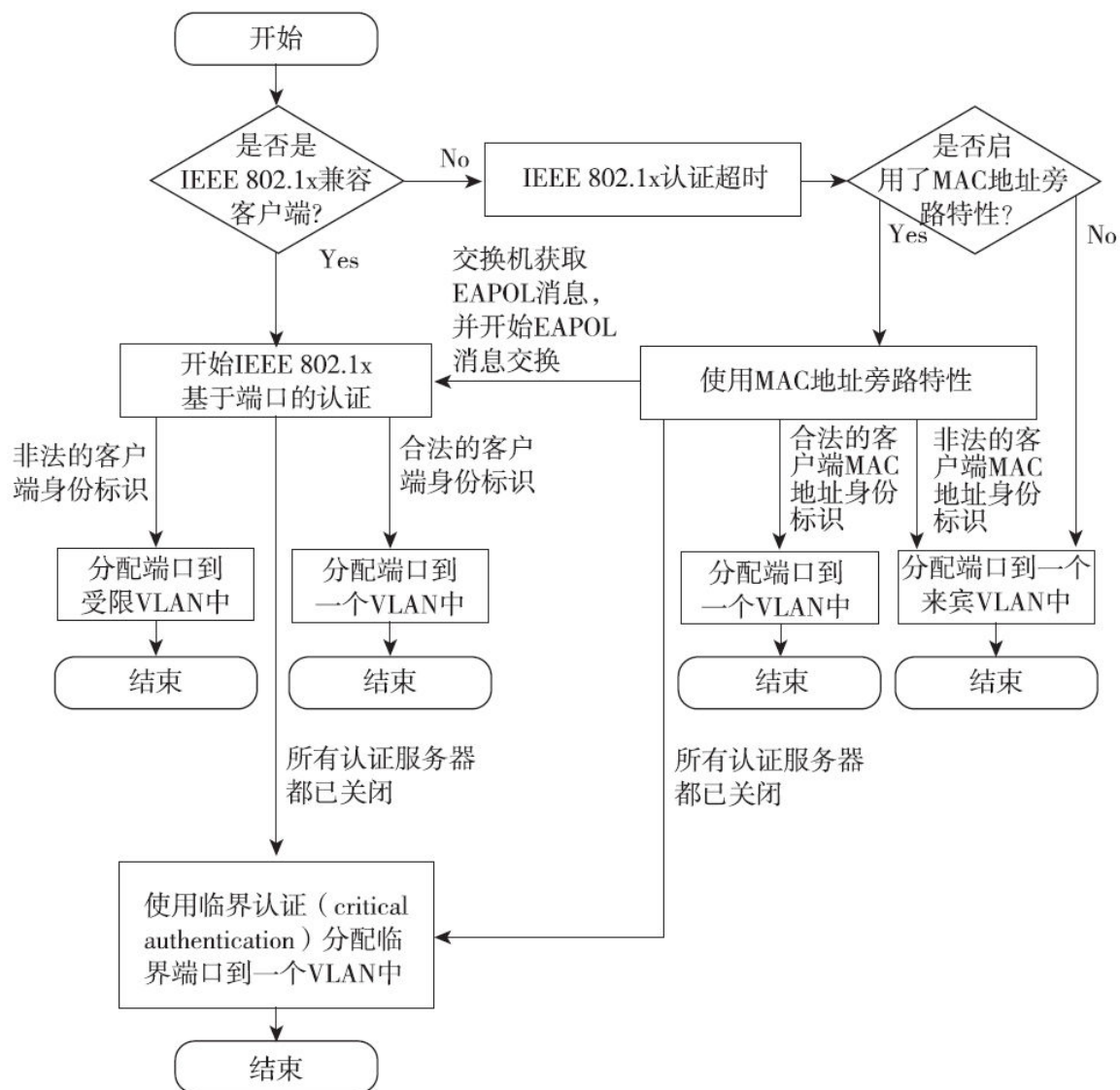


图 6-59 IEEE 802.1x认证流程

(1) 启用了周期性重认证 (Periodic re-authentication) 功能, 并且重认证计时器已过期。

可以使用一个指定的值, 或者RADIUS服务器上的配置值来配置重认证计时器。在使用配置的RADIUS服务器进行IEEE 802.1x认证后,

交换机使用基于会话超时（**Session-Timeout**）RADIUS属性（属性27）和终止行为（**Termination-Action**）RADIUS属性（属性29）的计时器。

会话超时属性指定了多长时间后进行重认证，而终止行为属性指定在重认证过程中将发生的行为。这个行为包括两种：初始化

（**Initialize**）和重认证（**ReAuthenticate**）。当设置为初始化行为时（此时默认属性值为**DEFAULT**），IEEE 802.1x会话终止，在重认证过程中断开连接；当设置为重认证属性时（此时属性值为**RADIUS-Request**），在重认证过程中IEEE 802.1x会话不受影响。

（2）通过**dot1x re-authenticate interface interface-id**特权模式命令对客户端进行重认证。

有关IEEE 802.1x协议的详细工作原理及配置请参见笔者编著的《Cisco/H3C交换机高级配置与管理技术手册》一书。

6.14 主要WLAN标准与技术

近年来，在局域网技术方面，最引人瞩目的应该非WLAN（无线局域网）技术莫属。可以说是一年一个跨越，短短10多年，从最初的2Mbps速率到了今天的600Mbps，现在千兆速率的WLAN标准也在开发，并即将发布之中。在其中不仅诞生了各种不同接入速率的WLAN接入规范，也诞生了许多相对应的新技术。本节就要对这些WLAN接入标准和相关主要技术作一个全面的介绍。

就像以太网中有许多以太网接入规范（在IEEE 802.3标准中）一样，WLAN也有许多对应的接入规范，它们均在IEEE 802.11标准之中。目前该系列包含以下四种WLAN接入规范包括：IEEE 802.11b、IEEE 802.11a、IEEE 802.11g和IEEE 802.11n，目前还有两种更高接入速率的WLAN接入规范正在研发之中，那就是IEEE 802.11ac和IEEE 802.11ad，它们的基本特性比较如图6-60所示。

规范号	IEEE 802.11b	IEEE 802.11a	IEEE 802.11g	IEEE 802.11n	IEEE 802.11ac	IEEE 802.11ad
发布时间	1999 年 9 月	1999 年 9 月	2003 年 6 月	2009 年 9 月	暂未发布	暂未发布
工作频段	2.4GHz	5GHz	2.4GHz	2.4/5GHz	5GHz	60GHz
非重叠信道数	3	12 或 24	3	15	8	暂未知
最高接入速率	11Mbps	54Mbps	54Mbps	600Mbps	3.2Gbps	7Gbps
频带	20MHz	20MHz	20MHz	20MHz/40MHz	20/40/80/160MHz	暂未知
调制方式	CCK/DSSS	OFDM	CCK/DSSS/OFDM	4*4MIMO-OFDM/DSSS/CCK	8*8MIMO-OFDM/16 ~ 256QAM	暂未知
兼容性	802.11b	802.11a	802.11b/g	802.11a/b/g/n	802.11a/b/g/n	802.11a/b/g/n/ac

图 6-60 IEEE802.11系列接入规范基本特性比较

WLAN采用的是IEEE 802.11系列规范，它也是由IEEE 802标准委员会制定的。1990年IEEE 802标准化委员会成立IEEE 802.11 WLAN标准工作组，最初的无线局域网标准是IEEE 802.11于1997年正式发布的，该标准定义了物理层和介质访问控制（MAC）规范。物理层定义了数据传输的信号特征和调制，工作在2.4000~2.4835GHz频段。这一最初的无线局域网标准主要用于难于布线的环境或移动环境中计算机的无线接入，由于传输速率最高只能达到2Mbps，所以业务主要被用于数据的存取。但随着无线局域网应用的不断深入，人们越来越认识到，2Mbps的连接速率远远不能满足实际应用需求，于是IEEE 802标准委员会推出了一系列高接入速率的新WLAN规范。

6.14.1 IEEE 802.11b规范主要特性

在WLAN的发展历史中，真正具有实用无线连接的WLAN标准还是1999年9月正式发布的IEEE 802.11b。该规范的主要特性如下：

（1）工作频段

IEEE 802.11b规范工作在免费的2.4GHz频段，室内有效传输距离为35m，室外有效传输距离为140m。

（2）传输速率

IEEE 802.11b规范的最高传输速率为11Mbps，还可根据实际网络环境调整为1Mbps、2Mbps和5.5Mbps（相对现在几百兆速率的WLAN来说，它早已被淘汰了）。

（3）调制方法

IEEE 802.11b规范可根据不同接入速率采用不同的调制技术：传输速率为1Mbps和2Mbps时，采用原来IEEE 802.11规范中的DSSS（Direct Sequence Spread Spectrum，直接序列扩展）、DBPSK（Differential Binary Phase Shift Keying，差分二相位键控）、DQPSK（Differential Quadrature Phase Shift Keying，差分四相位键控）等数字调制方法；传输速率为5.5Mbps和11Mbps时，采用CCK（Complementary Code Keying，互补编码键控）调制方法。

说明 DSSS是先将信号源与一定的PN（Pseudo Noise，伪噪声）码进行混合，然后通过DBPSK、DQPSK相位键控技术将原来信号中的0或1比特用11个chips的巴克序列（Barker sequence）：

{+1, -1, +1, +1, -1, +1, +1, +1, -1, -1, -1}进行替代，扩展频谱，使得原来较高功率、较窄频带的频率变成具有较宽频带的低功率频率。简单来说，DSSS利用高频率的信号，通过各个调变技术进行扩展频，将发送端的频谱信号频带展宽，而在接收端用相同的展频技术去进行译码，把展频后的信号还原成原始的信息。DSSS具有抗干扰能

力强、抗多径干扰能力强、对其他电台干扰小、抗截获能力强、可以同频工作、便于实现多址通信等优点。

（4）信道划分

IEEE 802.11b规范全球使用的是同一无线电模型，共有11个信道，每个信道带宽为22MHz，但相邻信道间只有5MHz带宽不重叠（也就是会重叠17MHz带宽），如图6-61所示。因为每两个相邻信道都会有大部分的频段重叠，所以在IEEE 802.11b规范的整个频段中，真正完全不重叠的信道最多只有3个，如图6-62所示。这些不重叠的信道有五种组合，那就是：1、6和11号信道，2和7号信道，3和8号信道，4和9号信道，以及5和10号信道。每个区域WLAN网络只能选择一组，且推荐使用1、6和11这组信道，这样就可以在同一个区域WLAN网络中部署3台AP了。为保证这部分区域所使用的信号信道不能互相覆盖，各AP要使用不同的信道，最好选择完全不重叠的一组信道。

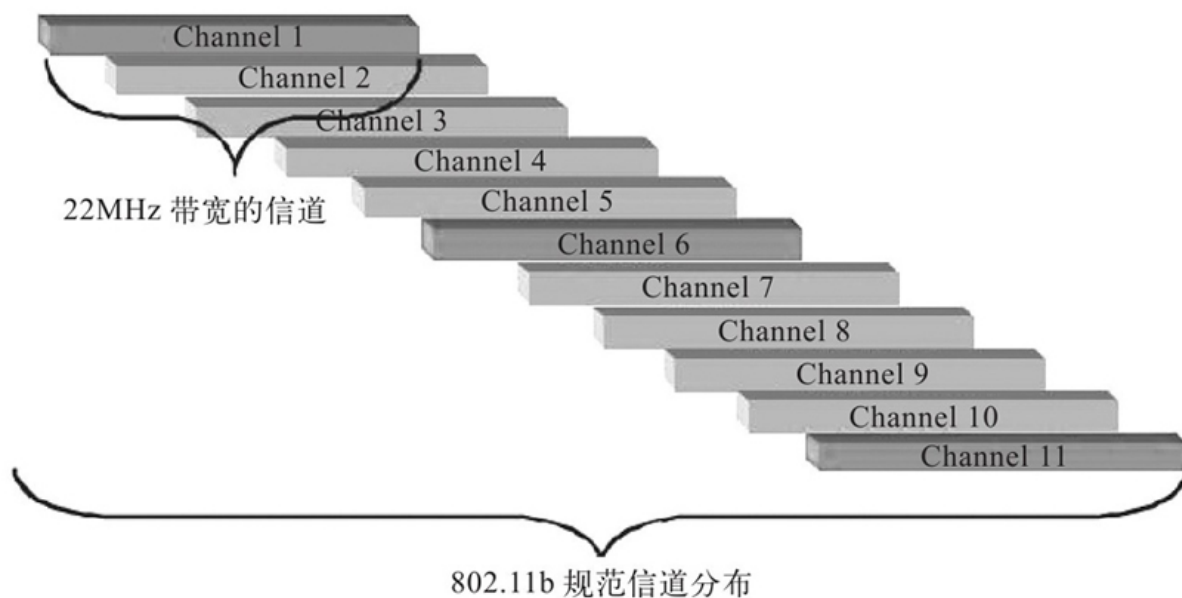


图 6-61 IEEE 802.11b信道分布

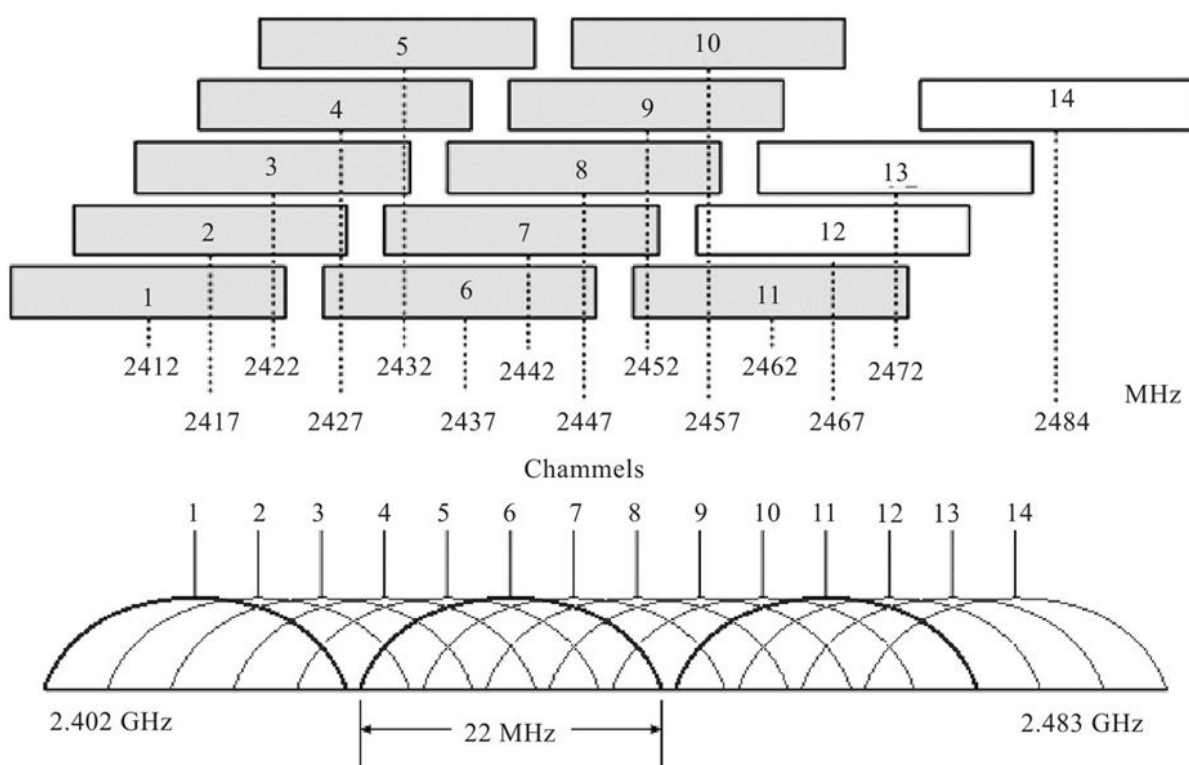


图 6-62 IEEE 802.11b规范信道划分及频率覆盖范围

IEEE 802.11b和IEEE 802.11g规范中的信道划分如表6-10所示。

表 6-10 IEEE 802.11b 和 IEEE 802.11g 规范中的信道划分

信道 ID	信道中心频率 /GHz	IEEE 802.11g 信道中心频率 /GHz
1	2.412	2.412
2	2.417	2.417
3	2.422	2.422
4	2.427	2.427
5	2.432	2.432
6	2.437	2.437
7	2.442	2.442
8	2.447	2.447
9	2.452	2.452
10	2.457（法国仅允许使用的频道）	2.457（法国仅允许使用的频道）
11	2.462（法国仅允许使用的频道）	2.462（法国仅允许使用的频道）
12	—	2.467（法国仅允许使用的频道）
13	—	2.472（法国仅允许使用的频道）

（5）主要安全技术

IEEE 802.11b规范主要采用的安全技术包括：SSID（Service Set Identifier，服务集标识符）和WEP（Wired Equivalent Privacy，有线等效保密）链路加密。在2003年以后生产的IEEE 802.11b规范的WLAN一般还支持WPA和IEEE 80.1x安全技术，但这通常是在同时支持IEEE 802.11b和IEEE 802.11g两种规范的设备中提供，单独的IEEE 802.11b设备不支持。

说明 SSID是一个无线局域网的名称。同一网络中的所有无线设备必须具有相同的SSID才能实现相互通信。无线客户端的SSID可以通过输入名称到网络设置进行手动设置，或者自动设置为不指定或空。

网络管理员通常使用公开的**SSID**，在**AP**中设置并广播给所有在网络范围内的无线设备。但现在新的**AP**默认是禁用了**SSID**自动广播这个功能的，以提高网络的安全性。另外，因为**WLAN**是通过电波进行数据传输的，存在电波泄漏导致数据被截听的风险，所以采用**WEP**链路加密保护措施。它是源自于**RSA**数据加密技术，提供了40位和128位长度的密钥机制，但它是一个对称加/解密方案，在数据的加密和解密过程中使用相同的密钥和算法，存在较大安全风险。

WPA（**Wi-Fi Protected Access**，**Wi-Fi**受保护访问）作为**IEEE 802.11i**标准的子集，包含了认证、加密和数据完整性校验三个部分，是一个完整的安全方案。其核心是**IEEE 802.1x**和**TKIP**（**Temporal Key Integrity Protocol**，临时密钥完整性协议）。

WPA是一种继承了**WEP**基本原理而又解决了**WEP**缺点的新技术，使用128位密钥。由于加强了生成密钥的算法，因此即便收集到分组信息并对其进行解析，也几乎无法计算出通用密钥。其原理为根据通用密钥，配合表示计算机**MAC**地址和分组信息顺序号的编号，分别为每个分组信息生成不同的密钥，然后与**WEP**一样将此密钥用于**RC4**加密处理。通过这种处理，所有客户端的分组信息所交换的数据将由各个不相同的密钥加密而成。无论收集到多少这样的数据，要想破解出原始的通用密钥几乎是不可能的。**WPA**还追加了防止数据中途被篡改的功能和认证功能。由于具备这些功能，此前**WEP**中备受指责的缺点得

以全部解决。WPA不仅是一种比WEP更为强大的加密方法，而且有更为丰富的内涵。

IEEE 802.1x是用于WLAN的一种增强性网络安全解决方案，就是用来控制无线工作站对AP的访问。如果认证通过，则AP为对应工作站打开逻辑端口，否则不允许用户上网。IEEE 802.1x要求无线工作站安装802.1x客户端软件，AP要内嵌802.1x认证代理，同时它还作为Radius客户端，将用户的认证信息转发给Radius服务器。IEEE 802.1x除提供端口访问控制能力之外，还提供基于用户的认证系统及计费，特别适合于公共无线接入解决方案。

6.14.2 IEEE 802.11a规范主要特性

虽然IEEE 802.11b规范的11Mbps传输速率比起规范的IEEE 802.11的2Mbps来说有了几倍的提高，但这也只是理论数值，在实际应用环境中的有效速率还不到理论值的一半。为了继续提高传输速率，IEEE 802工作小组继续了下一个规范的开发，那就是2001年底发布的IEEE 802.11a。

经验之谈 在这里要说明的一件事就是，为什么最先推出的规范命名为IEEE 802.11b，而后来推出的规范反而是IEEE 802.11a。那是因为，这两个规范是分属于两个不同的小组。事实上IEEE 802.11a规范与IEEE 802.11b的研制工作是同时开始的，只是在后来正式完成、发布中，IEEE 802.11b规范却走在了前面，所以最先发布的是IEEE 802.11b，而不是IEEE 802.11a。还有一点，那就是IEEE 802.11a规范本来要先于IEEE 802.11b发布，所以其速度原先的设想不是54Mbps，只是IEEE 802.11b发布了11Mbps的规范，所以IEEE 802.11a规范的连接速率就不可能再低于或者接近11Mbps，只能超过。

IEEE 802.11a规范的主要特性如下：

(1) 工作频段

IEEE 802.11a规范工作频段为商用的5GHz频段（不是采用IEEE 802.11b规范中的2.4GHz免费频段，所以不与IEEE 802.11b设备兼容），室内有效传输距离35m，室外有效传输距离120m。

（2）传输速率

IEEE 802.11a规范的最高数据传输速率为54Mbps，根据实际网络环境，还可调整为6Mbps、9Mbps、12Mbps、18Mbps、36Mbps、48Mbps。

（3）信道划分

IEEE 802.11a规范的每个信道的带宽有两种选择：20MHz或40MHz，如果为20MHz带宽，则共有24个不相互重叠的信道，如果是40MHz带宽，则共有12个不相互重叠的信道。表6-11列出了可用的24个信道及所适用的环境。

表 6-11 IEEE 802.11a 规范中的信道划分

信道 ID	信道中心频率 /MHz	适用环境
36	5180	室内
40	5200	室内
44	5220	室内
48	5240	室内
52	5260	室内或室外
56	5280	室内或室外
60	5300	室内或室外
64	5320	室内或室外
100	5500	室内或室外
104	5520	室内或室外
108	5540	室内或室外
112	5560	室内或室外
116	5580	室内或室外
120	5600	室内或室外
124	5620	室内或室外
128	5640	室内或室外
132	5660	室内或室外
136	5680	室内或室外
140	5700	室内或室外
149	5745	主要用于室外
153	5765	主要用于室外
157	5785	主要用于室外
161	5805	主要用于室外
165	5825	主要用于室外

(4) 调制方法

IEEE 802.11a规范采用52个OFDM（Orthogonal Frequency Division Multiplexing，正交频分复用）调制扩频技术，可提高信道的利用率。在52个OFDM中的52个载波中，48个用于传输数据，4个是引示副载波（pilot carrier，就是载波里面没有携带任何数据），每一个带宽为

0.3125MHz (20MHz/64) , 可以应用BPSK (二相移相键控) 、QPSK (四相移相键控) 、16-QAM或者64-QAM调制技术。OFDM技术将信道分成若干正交子信道, 将高速数据信号转换成并行的低速子数据流, 再调制到每个子信道上进行传输。正交信号可以通过在接收端采用相关技术来分开, 这样可以减少子信道之间的相互干扰。

(5) 主要安全技术

IEEE 802.11a规范在安全方面一开始也主要使用WEP加密技术和SSID。2003年以后生产的IEEE 802.11a规范的WLAN一般还支持WPA和IEEE 802.11i安全技术, 但这通常是在同时支持IEEE 802.11a和IEEE 802.11g两种规范的设备中提供, 单独的IEEE 802.11a设备不支持。

6.14.3 IEEE 802.11g规范主要特性

虽然IEEE 802.11a规范的速度已非常快了，但由于IEEE 802.11b与IEEE 802.11a两个规范的工作频段不一样，相互不兼容，致使一些原先购买IEEE 802.11b规范的无线网络设备在新的802.11a网络中不能用，于是推出一个兼容两个规范的新规范就成了事实之需，那就是2003年6月IEEE正式推出IEEE 802.11g规范。

IEEE 802.11g规范的主要特性如下：

(1) 工作频段

IEEE 802.11g规范与IEEE 802.11b规范一样工作在免费的2.4GHz频段（与IEEE 802.11b兼容，但不与IEEE 802.11a兼容），室内有效传输距离38m，室外有效传输距离140m。

(2) 传输速率

IEEE 802.11g规范具有与IEEE 802.11a规范一样的传输速率（54Mbps），总带宽为20MHz，如果需要的话，传输速率可降为48Mbps、36Mbps、24Mbps、18Mbps、12Mbps、9Mbps或者6Mbps。

(3) 信道划分

IEEE 802.11g规范共划分了13个信道，如表6-10所示，每个信道所覆盖的频率范围如图6-62所示，不同的只是IEEE 802.11g规范中12、13号这两个信道是可用的。在这13个信道中，真正完全不重叠的信道最多也只有3个，有五种组合，那就是：1、6和11（或13）号信道，2、7和12号信道，3、8和13号信道，4和9号信道，以及5和10号信道，每个区域WLAN网络只能选择其中一组，美国是推荐使用1、6和11这组信道，但世界上大多数国家还推荐使用1、5、9、13这组信道，这样就可以在同一个区域WLAN网络中部署4台AP了。为保证这部分区域所使用的信号信道不能互相覆盖，各AP要使用不同的信道，最好选择完全不重叠的一组信道。

（4）调制方法

IEEE 802.11g规范同时采用了IEEE 802.11a中的OFDM与IEEE 802.11b中的DSSS、CCK等多种调制技术。

（5）主要安全技术

在安全性方面，IEEE 802.11g规范全面支持IEEE 802.11i标准中的WPA、WPA2、EAP（Extensible Authentication，可扩展身份认证）AES(Advanced Encryption Standard，高级加密标准)加密，以及用于访问控制的IEEE 802.1x标准。

说明 在以上这三个主要无线局域网接入规范之外，我们还可见到诸如IEEE 802.11b+、IEEE 802.11a+和IEEE 802.11g+这三个所谓对应规范的增强版，它们的传输速度也相应增强，达到原有规范的2倍，分别为22Mbps、108Mbps和108Mbps。但这三个所谓的增强版规范并非正式的规范，而是一些无线网络设备开发商自己制定的企业规范，它们的兼容性较差，通常只能与本企业某些无线网络设备相兼容。

6.14.4 IEEE 802.11n规范主要特性

IEEE 802.11n，是2009年9月正式发布的IEEE新的802.11规范，也是目前最主要应用的WLAN接入规范。其主要特性如下：

(1) 工作频段

IEEE 802.11n规范可工作在2.4GHz和5GHz两个频段，所以它可以全面向下兼容以前发布的IEEE 802.11b/a/g这三个规范以。

(2) 传输速率

IEEE 802.11n规范在标准带宽（20MHz）单倍MIMO上支持的速率有7.2Mbps、14.4Mbps、21.7Mbps、28.9Mbps、43.3Mbps、57.8Mbps、65Mbps、72.2Mbps，使用标准带宽和4倍MIMO时，最高速率为300Mbps；在2倍带宽（40MHz）和4倍MIMO时，最高速率可达600Mbps，是最近的IEEE 802.11g的10倍多。

(3) 信道划分

IEEE 802.11n规范总共可以有15个不相互重叠的信道，其中在2.4GHz频段中有3个不相互重叠的信道，在5GHz频段中有12个不相互重叠的信道。另外，通过将两个相邻的20MHz带宽捆绑在一起组成一

个40MHz通信带宽，在实际工作时可以作为两个20MHz的带宽使用（一个为主带宽，一个为次带宽，收发数据时既能以40MHz的带宽工作，也能以单个20MHz带宽工作），这样可将速率提高一倍。同时，对于IEEE 802.11a/b/g，为了防止相邻信道干扰，20MHz带宽的信道在其两侧预留了一小部分的带宽边界。而通过频带绑定技术，这些预留的带宽也可以用来通信，从而进一步提高吞吐量。

（4）调制方法

IEEE 802.11n规范采用了IEEE 802.11g规范中相同的OFDM调制技术，只是选择的正交载波数更多。OFDM可将信道分成许多进行窄带调制信道和传输正交子信道，并使每个子信道上的信号带宽小于信道的相关带宽，用以减少各个载波之间的相互干扰，同时提高频谱的利用率。MIMO（Multiple-Input and Multiple-Output，多进多出）与OFDM技术的结合，就产生了MIMO OFDM技术，它通过在OFDM传输系统中采用阵列天线实现空间分集，提高信号质量，并增加多径的容限，使无线网络的有效传输速率有质的提升。

（5）主要安全技术

IEEE 802.11n规范与IEEE 802.11g规范所使用的主要安全技术类似，主要都是IEEE 802.11i所引入的WPA、WPA2和AES加密，以及IEEE 802.1x访问控制技术。

6.14.5 两个未正式发布的新规范简介

在IEEE WLAN接入规范中，目前还有两个新的规范正在研发，并且按计划都即将发布，那就是前面提到的IEEE 802.11ac和IEEE 802.11ad。因为最终的规范特性未正式发布，所以在此也仅作简单的介绍。

在我们还没有全面应用IEEE 802.11n规范的时候，新一轮的无线提速攻势又迎面扑来，那就是即将发布的IEEE 802.11ac和IEEE 802.11ad规范。与IEEE 802.11n规范不同的是，这两个新规范不再停留在百兆级别，而是要将整个WLAN也提高到了千兆领域，也将会兼容目前发布的全系列IEEE 802.11规范。

IEEE 802.11ac规范的核心技术主要基于IEEE 802.11a，继续工作在5.0GHz频段上以保证向下兼容性，但由于数据传输通道的大大扩充，IEEE 802.11ac规范的入门级速度就可以达到433Mb/s，高端设备会达到3.2Gb/s，使WLAN真正进入千兆时代。

IEEE 802.11ac规范继续用IEEE 802.11n规范的MIMO技术，但进行了增强。IEEE 802.11n规范的数据传输通道宽度为20/40MHz，最大能提供600Mb/s下载速度，而80MHz信道宽度的IEEE 802.11ac，在3x3

MIMO模式下就能提供1.3Gb/s的下载速度，如果信道宽度为160MHz，该速度则是高达2.6Gb/s。

虽然在2008年制定此规范时IEEE工作组就将目标定位在2012年，但由于IEEE 802.11系列规范一向都是先有鸡后有蛋，产品比规范早出的特点，所以目前我们实际上已经看到了主流网络设备厂商的实验性IEEE 802.11ac产品。如全球领先的芯片厂商Broadcom正式公布了IEEE 802.11ac芯片，随后包括华硕、贝尔金、D-Link、华为、联想等一系列网络厂商都相继声称将很快提供基于这一规范的WLAN产品。

如果前一个IEEE 802.11ac主要是用来接替IEEE 802.11a/g规范主要应用于企业WLAN中的话，那同时研发的千兆级IEEE 802.11ad规范则主要面向家庭娱乐视频、音频设备，且工作在与前面所有IEEE 802.11接入规范工作频段不同的60GHz频段上。IEEE 802.11ad规范的主要目标能够使得高清视频和无损音频成为可能，为家庭多媒体应用带来更完备的高清视频解决方案。

在技术上，IEEE 802.11ad规范仍然使用自IEEE 802.11n规范所引入的MIMO技术来实现多路传输，将使单一信道传输速率过1Gbps，最高传输速率可以达到7.2Gbps。IEEE 802.11n最多能支持4x4 MIMO，而IEEE 802.11ac则是支持8x8 MIMO，而802.11ad更是可以支持10x10 MIMO以上。

6.14.6 其他主要WLAN规范

除了以上介绍的几个当前主要的WLAN接入规范外，还有一些其他无线局域网规范。

1.IEEE 802.11e

IEEE 802工作组于2005年年底正式推出了IEEE 802.11e规范。该规范增强了原有的IEEE 802.11MAC信道接入方式，在原来的DCF（分布式信道功能）和PCF（点信道功能）基础上引入了一种新的接入机制HCF（Hybrid Channel Funcation，混合信道功能）。

HFC定义了两种服务质量提供机制：竞争性信道接入机制EDCA（Enhanced Distributed Channel Access，增强型分布式信道访问）与非竞争（中心控制）接入机制HCCA（HCF，Controlled Channel Access，混合协调功能控制信道访问）。简地说，EDCA和HCCA分别就是DCF和PCF的增强版，EDCA只能在竞争期内使用，提供了不同优先级的QoS，HCCA扩展了PCF，在竞争期和非竞争期内均可使用，提供了参数化的QoS。

EDCA指定了四种访问类型，每一种类型对应一类数据。每一个访问类别配置了四个参数，它们分别是：最小竞争窗口（Minimum

Contention Window, CW_{min}), 最大竞争窗口 (Maximum Contention Window, CW_{max}), 传送机会 (Transmission Opportunity, TXOP), 仲裁帧间间隔 (Arbitration Inter Frame Space, AIFS)。为每一类数据设置这些参数能够让网络管理员根据应用程序组合和通信量调整网络。

ECDA是对原有的DCF机制中的CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance, 载波侦听多路访问/冲突避免)的扩展, 用来保证对不同的数据流提供不同的QoS。与DCF相比, ECDA机制从两方面进行服务质量的区分: 一是基于退避机制, 为具有不同服务质量要求的数据流分配不同的CW_{min}和CW_{max}, 而DCF中对所有数据流的这两个参数是一样的; 二是基于仲裁帧间间隔AIFS, 也就是在接入信道前站点必须等待的信道空闲时间, 在DCF中, 每个站点必须等待的信道空闲时间均为DIFS (DCF帧间间隔), 而在EDCA中, 每个站点等待的时间是不同的, 由公式 $AIFS = SIFS + AIFSN * Q$ 来计算。通过分配不同的AIFSN值, 就可以获得不同的服务质量。

2.IEEE 802.11i

这是IEEE提出的新一代WLAN安全规范, 实际上是把1999年制定的原用于有线以太网的IEEE 802.1x安全规范引入了WLAN, 用于取代以前的WEP加密规范。它在加密处理中引入了TKIP (Temporal Key

Integrity Protocol，临时密钥完整性协议），使得用于链路加密的密钥从静态转变为动态。虽然还是基于RC4算法，但比采用静态密钥的WEP先进。除了密钥管理以外，它还具有以EAP（Extensible Authentication Protocol，可扩展认证协议）为核心的用户审核机制，可以通过服务器审核接入用户的ID，在一定程度上可避免了黑客非法接入。

IEEE 802.11i拥有如下的关键安全技术：

（1）暂时密钥完整性协议（TKIP）

TKIP负责处理无线安全问题的数据加密部分。TKIP是包裹在已有WEP密码外围的一层“外壳”，由WEP使用的同样的加密引擎和RC4算法组成。不过，TKIP中密码使用的密钥长度为128位，且它为每个数据所使用的密钥是动态变化的，安全性更高。

（2）CBC-MAC计数模式协议（CCMP）

CCMP（Counter mode with CBC-MAC Protocol，支持CBC-MAC协议的计数器模式）是IEEE802.11i中要求强制实现的新加密算法。CCMP在计数模式下采用了AES的算法，使用了CBC-MAC（Cipher Block Chaining Message Authentication Code，密码块链消息验证码）模式，以消息块作为运算基础，最后产生消息块认证码。在IEEE

802.11i规范中，CCMP使用了128位的密钥，可以对数据进行更加有效的加密保护。

(3) IEEE802.1x

IEEE 802.1x向受保护的网络提供了一个有效的身份验证和用户通信管理的框架，同时还能动态地改变密钥。IEEE 802.1x在有线和无线局域网媒介中都捆绑了可扩展身份验证协议（EAP），并支持多重身份验证。

(4) EAPOL

EAPOL（Extensible Authentication Protocol Over LAN，基于局域网的扩展身份验证协议）是802.1x协议定义的一种报文封装格式，主要用于在客户端和设备端之间传送EAP协议报文，以允许EAP协议报文在局域网中传送。

3.IEEE 802.11f

IEEE 802.11f规范解决了不同规范访问点（AP）之间的漫游通信问题，所用的主要协议是IAPP（Inter-Access Point Protocol，接入点间协议）。它主要解决IEEE 802.11系列各接入规范在网间互连方面存在的不足。用户在两个不同的交换网段（无线信道）或两种不同类型无

线网的接入点间进行漫游时，通过该协议可更好地维护网络连接，此时无线LAN具备蜂窝电话那样的灵活性显得至关重要。

4.IEEE 802.11h

IEEE 802.11h用于802.11a的频谱管理技术，增加了传输功率控制和动态频率选择。它力图在传输功率和无线信道选择上比IEEE 802.11a更胜一筹，主要在欧洲使用。

6.14.7 WLAN MAC帧格式

在WLAN体系结构中，MAC子层有两种工作方式：分布式协调功能（DCF）和点协调功能（PCF），但MAC子层帧结构均如图6-63所示。

2	2	6	6	6	2	6	0 ~ 2312	4字节
Frame Control	Duratiior ID	Address1	Address2	Address3	Sequence Contorl	Address4	Data	Frame Check Sum

图 6-63 WLAN/RM中的MAC帧结构

下面对各图6-63所示字段进行具体介绍。

1) Frame Control（FC，帧控制）字段

FC字段占2个字节，用于控制MAC子层帧信息和行为。在这个字段的2个字节中又包括如图6-64所示的结构。

2	2	4	1	1	1	1	1	1	1	1位
Protocol Version	Type	Subtype	To DS	From DS	More Flag	Retry	Pwr Mgt	More Data	WEP	Order

图 6-64 FC字段结构

□Protocol Version: 协议版本号字段, 占2位。表示IEEE 802.11规范版本。

□Type: 帧类型字段, 占2位。帧类型包括管理、控制和数据三种类型。

□Subtype: 帧子类型字段, 占4位。帧子类型包括认证帧 (Authentication Frame)、解除认证帧 (Deauthentication Frame)、连接请求帧 (Association Request Frame)、连接响应帧 (Association Response Frame)、重新连接请求帧 (Reassociation Request Frame)、重新连接响应帧 (Reassociation Response Frame) 解除连接帧 (Disassociation Frame)、信标帧 (Beacon Frame)、Probe帧 (Probe Frame)、Probe请求帧 (Probe Request Frame) 或Probe响应帧 (Probe Response Frame)。

□To DS: 到分布式系统的帧字段, 占1位。当帧是发送给 Distribution System (DS) 时, 该值设置为1。

□From DS: 来自分布式系统的帧字段。当帧是从DS处接收到时, 该值设置为1。

□More Fragment: 更多分片字段, 占1位。表示当前帧后面还有更多分段属于相同帧时, 该值设置为1, 否则设为0。

□Retry: 重传字段, 占1位。如是重传帧则用1表示, 否则用0表示。

□Pwr Mgt: Power Management, 电源管理字段, 占1位。表示在帧传输后, 站点所采用的电源管理模式。1表示采用节能模式, 0表示活动模式。

□More Data: 更多数据字段, 占1位。1表示在AP缓存中还有从分布式系统到节能模式站点的帧, 0表示没有。

□WEP: 加密字段, 占1位。1表示采用WEP (Wired Equivalent Privacy) 算法对帧数据进行加密, 0表示不加密。

□Order: 顺序字段, 占1位。1表示按顺序发送帧或者分段, 0表示不按顺序发送。

2) Duration/ID

Duration/ID为持续时间字段, 占2字节。当第15位为0时, 用于设置NAV (网络分配向量), NAV等于在当前传输中介质忙的时间 (以毫秒计) 长。这样所有站点就会监控接收到的帧送, 并更新NAV。

在IEEE 802.11 WLAN网络内, 所有接收到RTS (Request to Send, 请求发送) 与CTS (Clear to Send, 清理后发送) 信号的无线设备, 都将采用虚拟介质检测 (Virtual Carrier Sense, VCS) 机制, 设置NAV

（Network Allocation Vector，网络分配矢量），并使用在RTS和CTS中包含的Duration/ID字段信息来设置MAC参数NAV，Duration/ID字段指明了源和目的主机为传输数据将要占用信道的时间长度。当物理层内的NAV指针打开时，设备将认为此时的物理介质正被其他设备所占用而停止发送与接收数据。NAV的值随着时间推移不断减小，在NAV值减到零之前，主机不会发起传输尝试。

VCS机制设置使其他主机预先知道信道中正在进行的传输情况，从而有效提高了数据帧成功传输的概率。但是VCS机制增加了RTS和CTS的开销，降低了有效数据传输速率。NAV的设计有助于解决无线局域网内隐藏节点的问题。

在没有冲突发生时，第14位为0，第15位为1，所有其他位为0。这样得出的NAV值为32768。所有站点会在无冲突期间更新NAV值，以免发生冲突。

3) Address

Address为地址列表字段，包括图6-63中的4个地址（Address 1、Address 2、Address 3、Address 4，这里当然是MAC地址了）字段，它们依次对应：接收者地址、发送者地址、源地址和目标地址。每个地址字段占6字节（48位）。这4个字段对于所有MAC帧来说并不是都需要的，是否需要取决于帧类型。

当第1位为0时，表示该地址为单一站点所用的单播地址；当第1位为1时，表示该地址对应一组站点的组播地址；如果所有位均为1，则表示该帧为广播帧。

4) Sequence Control

Sequence Control为序列控制字段，占2字节。它是由分段号和序列号组成，用于表示同一帧中不同分段的顺序，并用于识别数据包副本。其中高4位表示分段号，从0开始计数，步长为1。后12位是序列号（也就是优先级号），用于决定以模为4096的传输帧计数器，也是从0开始的，步长为1。同一帧的分段，序列号是一样的。

5) Data

Data为数据字段。发送或接收的信息。最大的帧为2312字节，其中包括8字节的802.11 LLC头，加上2296字节的净负荷和WEP开销。如果此字段为空，则表示该帧为控制和管理帧。

6) Frame Check Sum

Frame Check Sum为帧校验序列，即CRC（Cyclic Redundancy Check，循环冗余校验），占4字节。用于校验帧的完整性，校验时必须对除FCS字段外的其他字段一起进行计算。

第7章 网络层

如果我们把物理层和数据链路层比作市内交通，那么本章所要介绍的网络层就可以比作连接不同城市交通的中转车站、机场或码头。就像中转车站、机场或码头可以把来自其他城市的旅客送到下一站或本市目的地一样，网络层可以把来自其他网络中的数据传送到下一个途经的网络或本网络中的目的节点。当源端和目的端位于不同网络的时候，直接通信是不可行的，此时就需要由网络层解决。

网络层（在TCP/IP体系结构中称为网际互连层）是网络体系结构中非常重要的一层，在技术上又是非常复杂的一层，因为它既要解决不同网络的节点间通信的路由和协议识别问题，又要通过路由选择策略解决网络拥塞问题，尽可能提高网络通信的可靠性。网络层关注的是如何将分组从源端沿着网络路径传送到目的端。为了实现这个目标，网络层必须知道通信子网的拓扑结构，并且在拓扑结构中选择适当的路径。同时网络层还必须谨慎地选择路由路径，以避免发生某些通信线路和路由器负载过重，而其他线路和路由器空闲的情形。

本章主要讨论网络层的主要功能、主要服务类型、主要路由算法、拥塞控制方法及原理、主要网络层通信协议报文格式和工作原理，以及三层交换原理。关于IP地址及子网划分将在第8章介绍，关于主要的动态路由协议及工作原理将在本书第9章介绍。

7.1 网络层概述

本书前面介绍的物理层和数据链路层构建了局域网内部的通信线路，相当于一个城市内部的交通路线。本章要介绍的网络层就是用来连接不同局域网线路的结点，位于不同网络的边缘，就相当于各城市边缘，用于连接不同城市交通线路的中转站点一样。

7.1.1 划分网络层的必要性

网络层是从功能上定义的一个逻辑层次，与物理层和数据链路层有具体的设备支持一样，网络层也有具体的设备来完成其相关任务，最典型的就是我们常用的路由器（**Router**）。路由器就相当于连接不同城市公路的中转车站，起数据中转作用，如图7-1所示。每个路由器至少可连接两个网络，就像一个城市的中转车站可以连接多个城市内的交通一样。

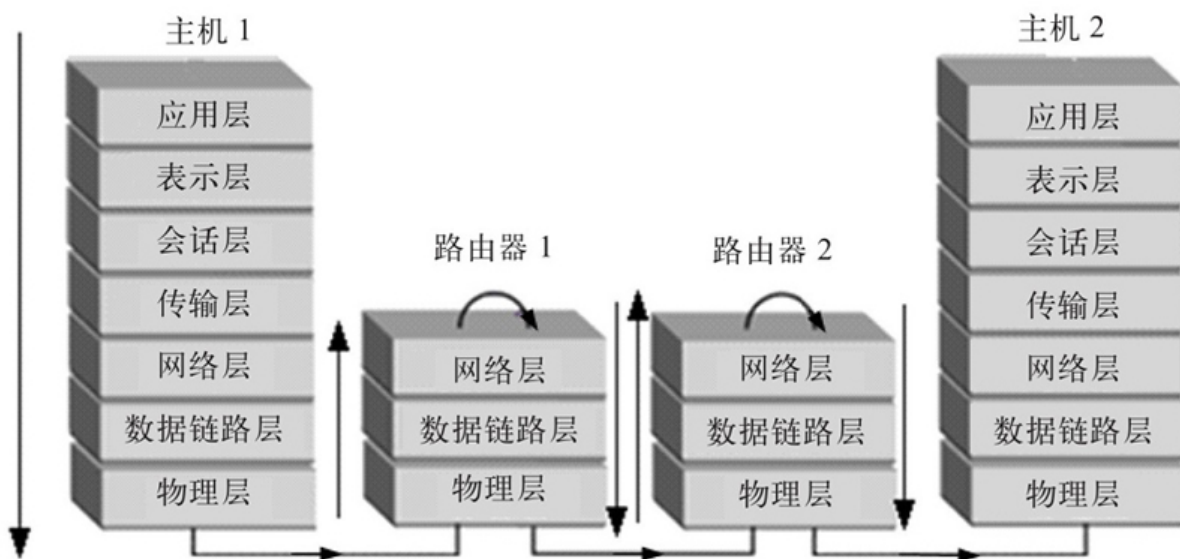


图 7-1 不同网络的连接示例

网络层是OSI参考模型中的第三层（对应TCP/IP协议体系结构中的第二层——网际互连层），介于传输层和数据链路层之间。总的来说，网络层的主要作用是实现两个网络系统之间的数据透明传送，具体包括路由选择、拥塞控制和网际互连等。网络层是端到端（也就是网络与网络之间）网络通信的最低层，在数据从数据链路层向传输层进行数据传输的通信中，起到构建一个中间通信子网的作用。它负责与它上面的资源子网（OSI/RM参考模型传输层及以上的四层）联系，是OSI/RM七层参考模型中面向网络通信的低三层（也即“通信子网”）中最为复杂、关键的一层。

网络层是计算机网络发展的产物，但并不是有了计算机网络就有了它。在早期的计算机网络中，基本上是以独立的局域网形式存在，

而我们知道，在局域网内部完全可以通过由物理层和数据链路层共同构建的通信链路来实现各计算机用户之间的通信访问，所以也就无需其他层次，包括网络层。况且当时计算机网络的应用比较有限，各局域网之间也不存在相互通信的必要性，所以这些局域网也就无须相互连接。但随着计算机网络的普及和发展，人们越来越发现，非常有必要把这些一个个孤立的局域网连接起来，组成一个更大的计算机网络，这样计算机网络的作用更能显现，可使更多人共享服务器和硬件资源。这就涉及计算机网络间的互连问题了。

那为什么要这样一个网络层呢？其实道理很简单，就是因为不同网络有不同的网络层协议和地址规范，一个网络中的用户若不能识别其他网络的通信协议和地址规范，就不能把数据从一个网络传送到另一个网络中。就像不同城市有不同的交通法规，属于不同的交警系统管理，不允许外地车辆随便出入一样，不同网络也有不同的设计规范，属于不同的组织来管理，必须通过授权，并由专门的协议来负责网络间的通信。

通常一个计算机网络就是一个管理边界，一般是属于一个特定的公司，由一个特定的管理者负责。所以在进行计算机网络互连时，要同时考虑两方面的问题：一是授权用户可以在不同网络间互访，共享双方的资源；另一方面又要保持各计算机网络管理原来的独立性，所以不能简单地通过拉一条网线就把问题解决了（这样无法解决管理独

立性问题)。事实上,在许多环境中,一般企业是不可能把两个位于不同城市甚至不同国家的计算机网络通过拉线的方式互连起来的。

在第6章中讲到,局域网内部的用户访问也是需要寻址的,即通过MAC地址(又称硬件地址)进行。但MAC地址属于数据链路层地址,不能跨网进行寻址,那么在不同网络间进行访问时又是通过什么地址来进行寻址呢?那就是网络层的功能了。在网络层也有一种对应的地址,即网络地址,每个网络都通过其网络地址,即NSAP(网络服务访问点)来标识,网络中的每个节点都有一个NSAP。这个NSAP就是由对应网络所运行的网络层通信协议来定义的。在目前最常见的TCP/IP协议网络中,这个协议就是IP协议,对应的NSAP就是IP地址。本书仅以TCP/IP网络(其实也就是TCP/IP协议体系结构中的网络互连层)为例进行介绍。有关IP协议和IP地址方面的内容将在第8章介绍。

在前面已说到,在物理层传输的是一个比特位(bit),在数据链路层中传输的是一个以许多字节为单位的帧(Frame),在每个帧的帧头都有源节点的MAC地址和目的节点的MAC地址,局域网内部的寻址就是通过MAC地址进行的;而在网络层中传输的是数据包

(Packet, 又称分组), 一个数据包是一个数据帧经过网络层协议重封装后得到的。每个数据包的包头都有源节点的IP地址和目的节点的IP地址,网络间的寻址就是通过IP地址进行的。在网络间的通信中,在网络体系结构中,数据是自上而下传输的(从网络层到物理层的数据单

位依次是包、帧和比特），接收方是自下而上传输的（从物理层到网络层的数据单位依次是比特、帧和包），如图7-2所示。

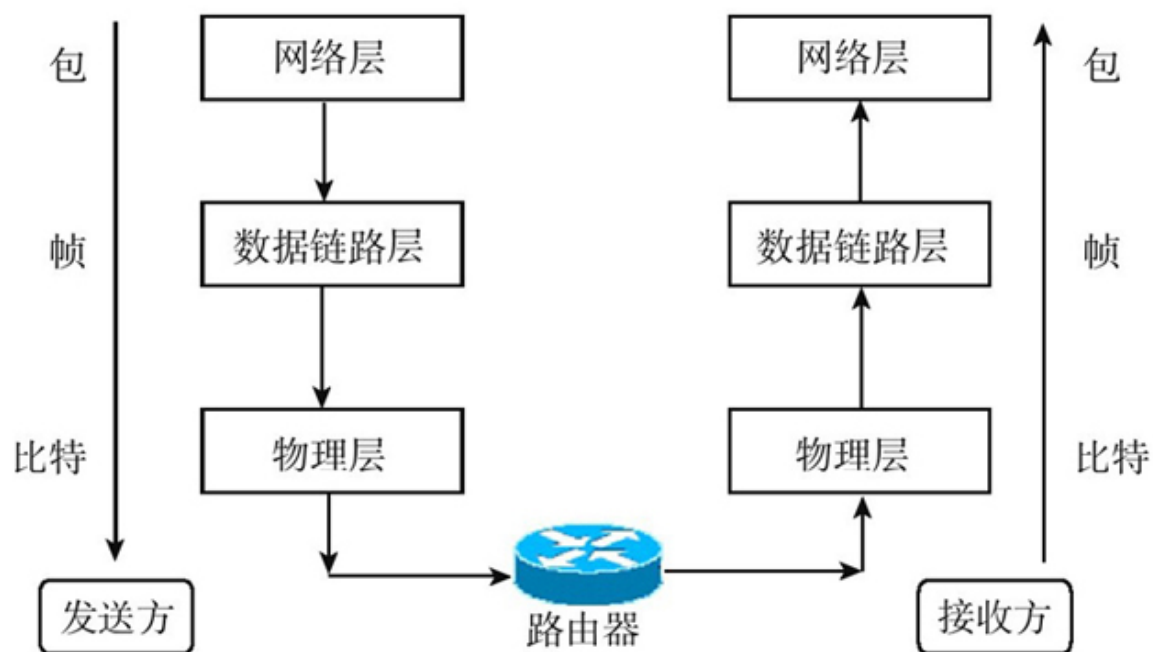


图 7-2 不同网络间主机通信的各层数据传输单位

7.1.2 网络层主要作用

网络层是为它的上一层——传输层服务的，并接受它的下一层——数据链路层所提供的服务。网络层的主要作用表现在以下几个方面。

(1) 屏蔽网络差异，提供透明传输

因为不同网络有不同的规范要求，就像不同城市有不同的交通规范一样。网络层就是为了解决这种不同差异，寻找一个不同网络间能共同遵守的网络通信规范，以便不同网络间能相互识别，并接受对方的访问请求。这样传输层就可以在不同网络间进行透明（也就是不管不同网络间的差异，就像通信双方是直接互连一样）数据传输了。

网络层向传输层提供的服务有两类：面向连接的网络服务和无连接的网络服务。虚电路服务是网络层向传输层提供的一种面向连接的服务，是可以使所有数据包按顺序到达目的节点的可靠数据传送方式。在虚电路服务中，进行数据交换的两个结点之间存在着一条专为它们服务的虚电路（相当于专线连接，或者点对点连接），无须附加网络地址。如我们所使用的各种拨号WAN连接、数据专线等接入方式都是属于面向连接的虚电路服务方式。

数据报服务是网络层提供的一种无连接的网络服务，只能提供不可靠的数据传送方式，源节点（如源路由器）发送的每个数据包都要附加网络地址、包序号等信息；目的节点收到的数据包不一定按序到达，还可能出现数据包丢失的现象。IP协议是一种无连接的网络层协议，提供无连接的网络层服务。

有关虚电路服务和数据报服务都将在本章后面具体介绍。

（2）为网络间通信提供路由选择

路由选择又称路径选择，是根据一定的原则和路由选择算法在多个结点的通信子网中选择一条到达目的节点的最佳路径的过程。确定路由选择的策略称为路由算法。在无连接的数据报服务中，网络结点要为每个数据包做出路由选择，就像你到达另一个城市时并不清楚该市的交通路线，所以只能在中转车站查看对应的交通线路来查找一个到达你要去的目的地最佳线路；而在面向连接的虚电路服务中，在建立连接时就已确定了路由路径，就像你坐专车直达目的地一样，根本无须选择旅行的路线。

（3）数据包封装和解封装

网络层要面临数据封装和解封装的问题，因为在网络体系结构的不同层次中传输的数据单位并不一样，且在发送方数据从上向下每经过一层都必须在数据头部添加对应层的协议头和（或）协议尾信息，

而到了接收方时，数据自下而上每经过一层时又都必须解除前面那层所封装的协议头和（或）协议尾信息。在发送方，来自传输层的报文通常是已根据对应网络链路的MTU（最大传输单元）被分为多个数据段，然后在网络层中对这些数据段头部添加一些网络层协议控制信息就组成了数据包，这就是包的封装过程。数据包的头部包含源节点和目标节点的网络层地址（如IP地址）。在接收方，数据从低层到达网络层时，要去掉在数据链路层加上的数据链路层协议控制信息（也就是帧头和帧尾），还原出原来的数据包格式，这就是包的解封装过程。

（4）拥塞控制

拥塞控制是为了避免网络传输路径中数据的传输延迟或死锁。在数据链路层我们提到了流量控制功能，那是针对数据链路中点对点传输速率的控制，而这里的拥塞控制是针对在网络传输路径中的端到端传输效率的控制。在网络层进行拥塞控制时主要采用预约缓冲区、许可证和分组丢弃等方式，具体将在本章后面介绍。

7.2 网络层数据交换及相关技术

数据到了网络层后，路由器是如何把这些数据包（或数据分组）转发到位于另一网络中的目的结点呢？这就要涉及网络层的数据交换技术了。从整个网络层数据交换技术的发展来看，交换技术经历了一个由线路交换、报文交换到现在最常用的分组交换的历程。报文交换和分组交换都属于存储-转发交换技术。

在计算机网络中，两个端点之间通常需要通过中间结点实现数据通信，这些中间结点并不关心数据内容，只提供一个交换设备，把数据从一个结点转发到另一个结点，直至达到目的端。数据交换技术主要是指网络中间结点所提供的数据交换功能。

7.2.1 线路交换

线路交换（Circuit Switching，又称电路交换）是最原始的数据交换方式，是在网络中利用可切换的物理通信线路直接连接通信双方所进行的一种数据交换方式。最常见的例子是电话交换系统和ISDN（Integrated Services Digital Network，综合业务数字网）系统。

线路交换是面向连接的服务，两台计算机通过通信子网进行数据交换之前，首先要在通信子网中建立一个实际的物理线路连接（通常

由一种开关电路来控制，如图7-3所示）。线路交换的最主要特点就是在进行数据交换前需要在一对用户之间建立起一条专用的数据通路，在整个数据传输过程中要经过线路建立、数据传输与线路释放这三个阶段。

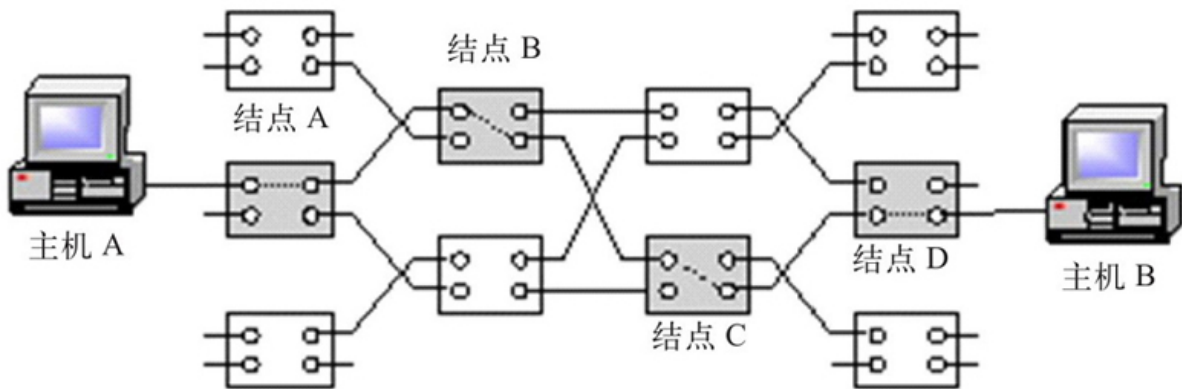


图 7-3 线路交换物理线路连接示意图

□线路建立：通过呼叫完成逐个结点的连接过程，建立起一条端到端的直通物理线路。

□数据传输：线路建立好后就可以直接在端到端的直通线路上传输数据。

□线路释放：数据传输完成后，由任一用户向交换网发出释放请求信令。该信令沿通路各结点传送，指挥这些结点拆除对应的链路，以释放信道资源。

图7-4列出了线路交换的三个主要过程以及每一过程中的主要步骤。从图中可以看出，在线路交换方式的线路建立和线路释放这两个主要阶段中，都需要由一方发送请求分组，然后另一方确认后返回应答分组，这是一个二次握手的过程。在数据传输这个阶段中，一方也是需要返回确认应答的，否则会在重传定时器超时后重传对应的数据。数据传输部分与传输层的对应功能类似，具体参见第10章。

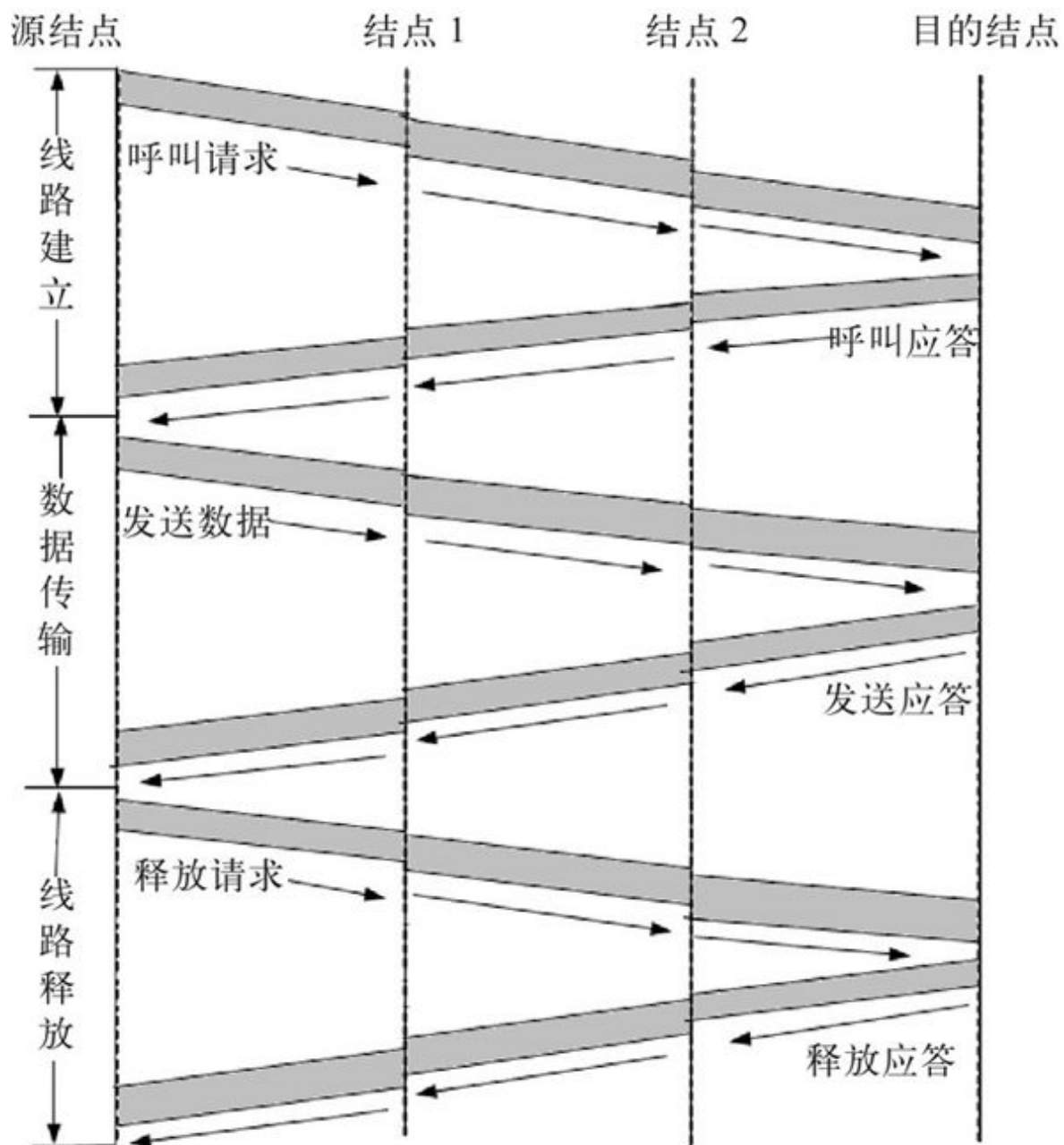


图 7-4 线路交换的三个主要过程

线路交换方式的优点是：通信实时性强，适用于交互式会话类通信。其缺点是：对突发性通信不适应（因为需先建立好的物理连

接)，整个数据交换系统效率低，系统也不具有存储数据的能力，很难实现拥塞控制。

7.2.2 存储-转发

存储-转发（Store-and-forward）从其名字就可以看出，这种数据交换方式是网络结点运用程序先将途径的数据流按传输单元（可以是报文或报文分组）接收并存储下来（同检验该数据单元的校验和），一个数据单元接收完后根据相关的路由算法选择一条合适的路由路径将数据转发出去，在逻辑（不是物理线路）上为数据流提供了传输通路。

存储-转发交换方式与线路交换方式相比具有如下特点：

□发送的数据要与目的地址、源地址、控制信息一起按照一定格式组成一个数据单元（报文或报文分组）进入通信子网，如图7-5所示。

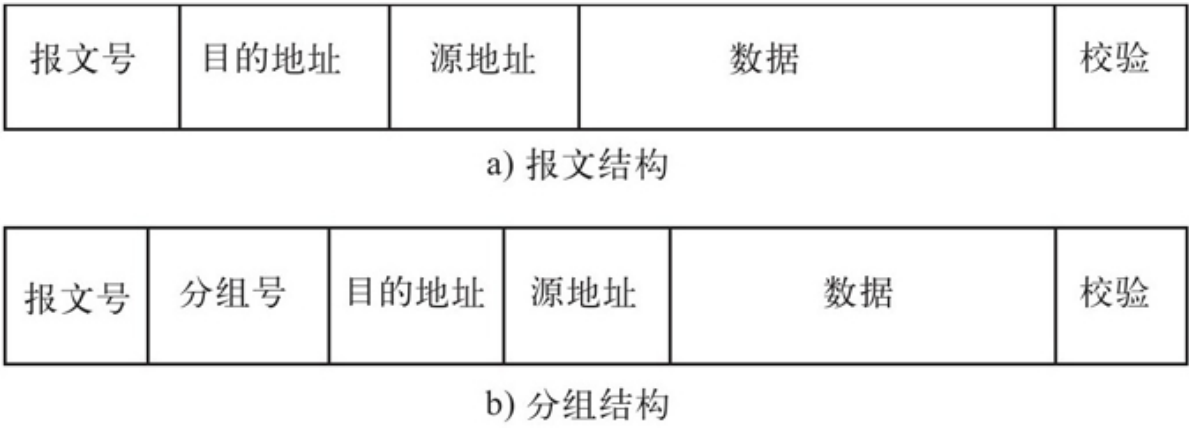


图 7-5 报文及报文分组结构

说明 数据通过通信子网传输时可以有报文（message）与报文分组（packet）两种方式。报文传输方式是不管发送数据的长度是多少，都把它当作一个逻辑单元发送；报文分组传输方式是限制一次传输数据的最大长度，如果传输数据超过规定的最大长度，发送结点就将它分成多个报文分组发送。

由于报文分组长度较短，所以在传输出错时检错容易并且重发花费的时间较少；另外，由于限定分组最大数据长度，有利于提高存储-转发结点的存储空间利用率与传输效率。公用数据网采用的是分组交换技术。但要注意，在报文分组中也有两种格式，原始的分组是不带地址信息字段的，而数据报分组是带地址信息字段的。所以图7-5b更准确地讲是数据报分组格式。

□通信子网中的结点是通信控制处理机（如路由器、三层交换机），其负责完成数据单元的接收、差错校验、存储、路选和转发功能。

存储-转发数据交换方式的优点如下：

□通信子网中通信控制处理机具有路由功能，可以动态选择报文分组通过通信子网的最佳路径；

□可以有效地进行拥塞控制，提高端到端系统传输效率；

□数据单元在通过通信子网中的每个通信控制处理机时，均要进行差错检查与纠错处理，因此可以减少传输错误，提高系统可靠性；

□通过通信控制处理机可以对不同通信速率的线路进行转换，也可以对不同的数据代码格式进行变换。

根据所传输的数据单元是报文还是报文分组，存储-转发交换方式又可细分报文交换和分组交换两种，下面具体介绍。

1. 报文交换

报文交换（**Message Switching**）是指信息以报文（**Message**，完整数据的一个信息段）为单位进行存储-转发的一种数据交换方式。在报文交换方式中，报文是网络中交换与传输的数据单元，即站点一次要发送的数据块，其包含了将要发送的完整的数据信息，其长短可能不一致，长度不限且可变。所谓存储-转发是当报文到达路由器后先存储起来，等待路由器分配资源再进行数据分组的转发。

报文交换的原理是用户发送的数据不是直接发送给目的节点的，而是先在中间结点上进行缓存（这类中间结点通常是由具有存储能力的交换机、路由器承担），然后再由中间结点在线路空闲时把数据发送出去。如图7-6所示的是一个报文交换示例。其中中间结点1是用来缓存站点A发送的数据的，而中间结点3是用来缓存站点B发送的数据的。

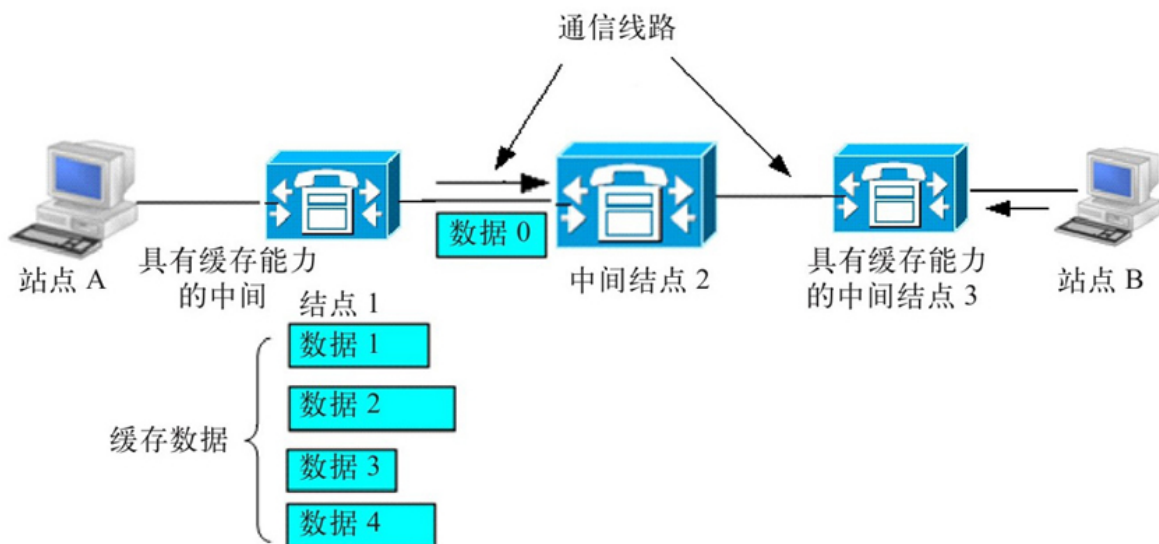


图 7-6 报文存储转发示例1

报文交换无须同时占用整个物理线路。如果一个站点希望发送一个报文，就将目的地址附加在报文上，然后将整个报文传递给中间结点；中间结点暂存报文，根据地址确定输出端口和线路，排队等待，当线路空闲时再转发给下一结点，直至终点。

目前在多数情况下网络中的每个结点都有存储功能，都可以将完整地接收的报文先暂存起来，然后将报文发送到下一个更接近目的主机的结点中。如此操作，直至将报文发送到目的主机为止。由此可以看出，它采用的就是数据报服务方式。图7-7所示的是另一个报文存储转发示例，报文首先由源主机 H_A 发出，到达最近的路由器 R_3 ， R_3 边接收边存储，待全部接收完后 R_3 再根据路由选择把该报文转发到 R_4 ， R_4 同样采取存取-转发方式，然后再根据路由选择把该报文转发到 R_5 ， R_5 采取同样的方式，最终把报文一一转发到目的主机 H_B 。

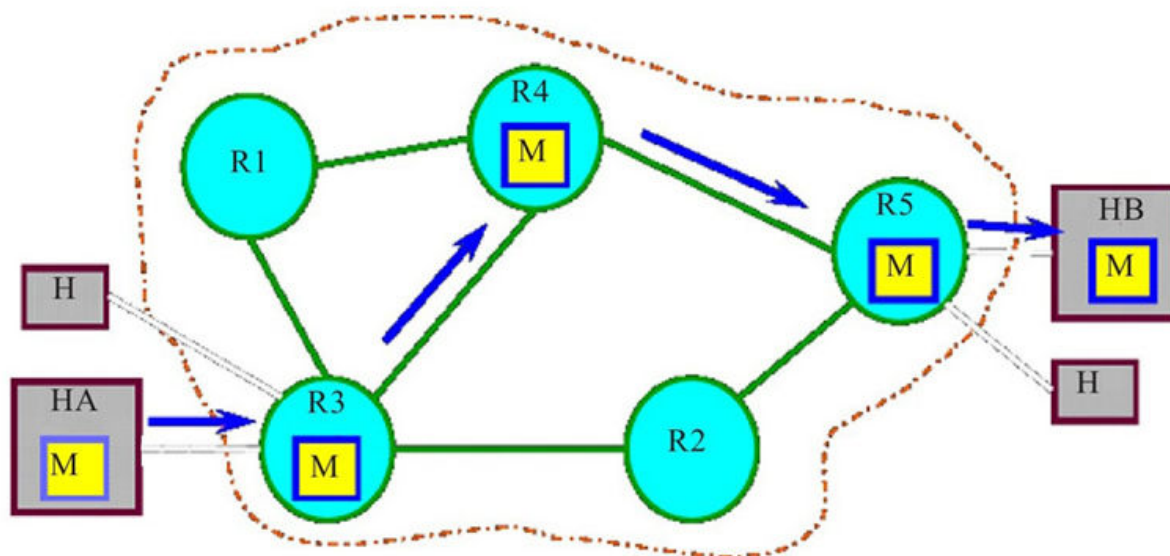


图 7-7 报文存储转发示例2

2.分组交换

分组交换（**Packet Switching**，也就是包交换）技术。在分组交换技术中有两大技术派系：一种是采用路由技术（也是目前比较普遍采用的一种交换方式），在要转发的数据包头部加上源节点和目的节点的IP地址，然后通过路由技术一级级地把数据转发下去。这种分组交换技术就是通常所说的数据报服务，其中的分组称为数据报

（**Datagram**）。

另一种是不依靠路由技术，而是在进行数据分组转前先在源节点和目的节点间的所有路由器间建立一条虚拟的通信通道，然后再把数据分组从这个虚拟通道中转发到目的节点。这种分级交换技术就是虚电路服务，所建立的虚拟通道称为“虚电路”（**Virtual Circuit, VC**）。

采用虚电路方式进行分组交换的典型实例就是通过ADSL拨号连接ISP网络（我们并不需要配置好到达ISP网络的路由），在配置时我们就要配置好ISP（Internet服务商）的VC值。

分组交换是结合报文交换和线路交换两种交换方式的优点而新开发的一种数据交换方式。分组交换也采用报文交换的存储-转发机制，但是规定了传输数据的单位长度，过长的报文被分成较小的单位（**Packet**，分组），依次发送。现在主要采用这种数据交换方式。

通过前面的介绍我们已经知道，每个报文会有完整的发送信息，包括源和目的地址，但是如果把一个报文分成了几个分组，这些中间的分组肯定是不包含有完整的发送信息的，那这些分组又是如何正确传输的呢？这时就要根据上节介绍的两种服务方式来选择：通过数据报这种服务方式为每个分组都添加了报文号、分组号、目的地址、源地址和校验字段信息（如图7-5b所示），然后将这些信息发送出去，由通信子网中的结点进行路由选择。在一个报文的所有分组到达了目的主机后，再将各个分组按照序号编排起来。

采用虚电路这种服务方式的分组无须添加源和目的地址信息，但仍需要添加报文号、分组号信息，因为在发送任何分组之前，首先在发送主机和目的主机之间建立一条逻辑连接（也就是建立一条虚电路，无须进行路径选择），然后所有的分组都将按照顺序依次被发送到目的主机。在所有的分组都发送之后，虚电路将被拆除。每台主机

可以和另一台主机建立若干个虚电路，每台主机也可以同时和若干台主机建立虚电路。

有关数据报和虚电路这两种服务方式将在下面两节具体介绍。

7.2.3 虚电路分组交换

上节已介绍了，在分组交换中又有两种可选的工作方式，那就是数据报方式和虚电路方式，对应向传输层提供数据报服务和虚电路服务。对于面向连接的服务（如拨号通信）则需要采用虚电路服务进行分组交换。就像数据链路层在传输数据前需要先建立链路连接一样，在虚电路分组交换中，分组被发送之前，必须在发送方与接收方之间建立一条专用的逻辑连接（虚电路，VC），并且以一个在所有经过的节点或结点上均唯一的虚电路标识符（Virtual Circuit Identifier，VCI）进行标识。这条VC所代表的就是所有经过的节点和结点的串连。

从某个结点到其他结点间可能有无数条虚电路存在，用于支持这两个端系统之间的不同数据传输。一个结点也可以同时与多个结点之间具有虚电路，每条虚电路支持特定的两个结点之间的数据传输。在虚电路分组交换中，为了进行数据传输，网络的源节点和目的节点之间要先建一条逻辑通路。每个分组除了包含数据内容外，还包含一个虚电路标识符。在预先建好的路径上的每个结点都知道把这些分组引导到哪里去，不再需要路由选择判定。最后，由某一个站用清除请求分组来结束这次连接。

在图7-8所示的示例中建立了四条虚电路，分别用于PC1与PC3，PC1与PC4、PC2与PC3、PC2与PC4的通信，它们所对应的VC分别是PC1-R1-PC3、PC1-R1-R3-PC4、PC2-R2-R1-PC3、PC2-R2-R3-PC4。每条VC创建后均以所有经过的节点和结点唯一的标识符号（从1开始）自动创建一个VC号。

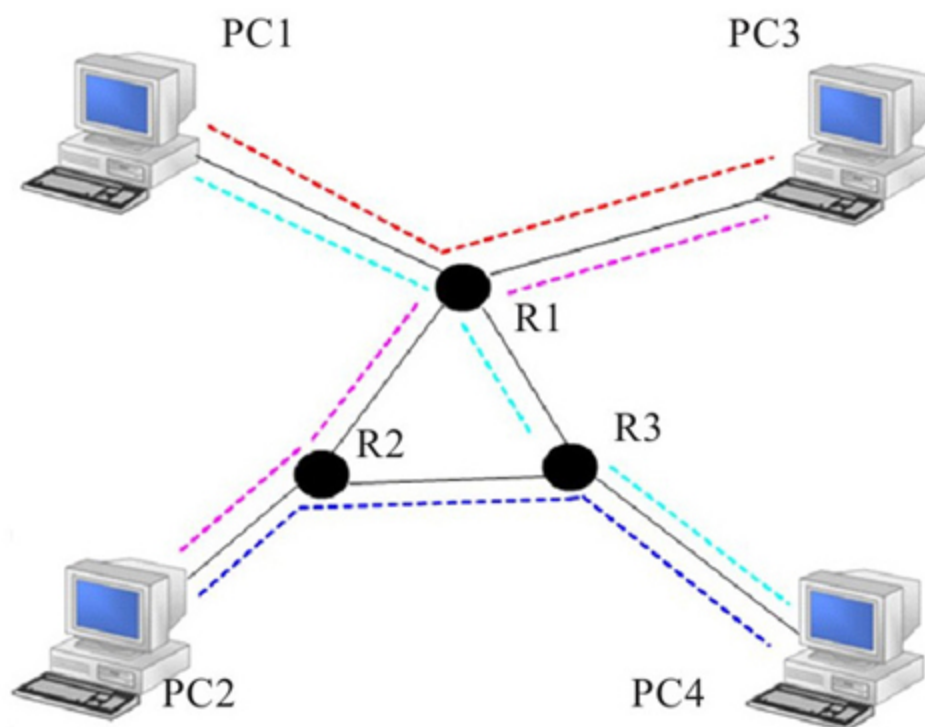


图 7-8 “虚电路”示例

在如图7-8所示的示例中，假设R1中已创建了从PC1到PC3以及从PC2到PC3这两条VC，并且所分配的VC号分配为VC1、VC2。R3中已创建一条由PC2到达PC4的VC，假设标识为VC3。如果PC1再要通过R1、R3创建一条到达PC4的VC，那么，尽管PC1中当前只有一条

VC1，但是R1中已有两条VC，即VC1和VC2，R4中也只有一条VC，即VC3，但是此时所新建的VC号只能大于3（如VC4、VC5，……），因为在新建的VC路径中各节点和结点已用到了VC1、VC2、VC3这三个标识符。返程应答路径与数据发送路径相反，但仍采用同一条VC通道进行传输，如PC1向PC4发送数据的VC路径是PC1-R1-R3-PC4，假设所分配的VC标识符号为VC1，而PC4向PC1应答的信息传输路径为PC4-R3-R1-PC1，但仍使用VC1这条虚拟通道。

在虚电路分组交换中，一次通信的所有分组都通过所创建的这条虚电路路径进行传输（而不会像在数据报分组交换中，不同分组可能沿不同路由路径进行传输），分组也不必带目的地址、源地址等辅助信息。而且，虚电路交换方式与线路交换方式类似，在开始进行数据交换前，也是需要先建立好线路连接，只不过线路交换中建立的是物理连接，而在虚电路交换中建立的是逻辑连接。

整个虚电路交换过程分为以下三个阶段：虚电路建立、数据传输与虚电路释放。整个过程可参见图7-4所示，只不过此处首先建立的是虚电路连接，而不是物理连接，另外，所发送的不是整个数据报，而是报文分组。

（1）建立虚电路

网络的源节点和目的节点之间要先建立一条逻辑通路。在图7-8所示的示例中，假设PC1有一个或多个报文要发送到PC4去，那么PC1首先要发送一个呼叫请求分组到结点R1，请求建立一条到节点PC4的连接。结点R1根据它的拓扑连接情况确定到达目的节点PC4最佳下一个结点为R3，与之建立起逻辑连接后把来自PC1的呼叫请求分组转发给R3，结点R3最终把呼叫请求分组转发到目的节点PC4。如果PC4准备接收这个连接请求，就发送一个呼叫应答分组到结点R3，然后通过上述相反路径把呼叫应答分组通过结点R1返回到PC1。这样就在源节点PC1和目的节点PC4之间建立起了一条逻辑通路PC1-R1-R3-PC4，然后根据这条路径中所有节点和结点中已分配的VC标签的情况，为该条新建的VC分配一个对于路径中各节点和结点均唯一的VC标识符，完成虚拟电路建立过程。

（2）传输数据

在逻辑通路建立好后，就可以在虚电路上交换数据了。每个分组除了包含数据部分外，还包含所采用的VC号，这样既可以确保这些分组选择正确的VC通道进行数据交换，又可以使各结点识别这些分组是否属于同一次通信内容。然后根据这个VC号所对应的路径把这数据分组依次传送到目的节点，不再需要进行路径选择。

（3）拆除虚电路

当本次数据全部传输完后，其中任意一个结点均可发送拆除虚电路的请求分组来结束这次连接，其他结点收到这个分组后立即终止当前所用的这条虚电路，释放该条虚电路所占用的系统资源。

综上所述可以得出，虚电路分组交换方式具有以下几个主要特点：

- 在每次分组交换前，必须在发送方与接收方之间建立一条逻辑连接。

- 一次通信的所有分组都通过这条虚电路顺序传送，也就相当于有固定的交换路径，所以每个分组均不必带目的地址、源地址等信息。

- 分组通过虚电路上的每个结点时，结点只需要做差错检测（做校验和检测），而不需要做路径选择。

- 通信子网中每个结点可以和任何结点建立多条虚电路连接，每条物理线路可以建立无数条虚电路连接，每条虚电路支持特定的两个结点之间的数据传输。

7.2.4 数据报分组交换

分组交换除了上节介绍的虚电路交换方式外，还有一种应用更为普遍的数据报交换方式。在虚电路分组交换方式中，分组数据是通信逻辑虚电路信道进行传送的，而在数据报分组交换方式中，每个分组数据均被视为一个数据报，可以单独通过路由路径发送，无须先创建传送的虚电路，但网络结点要为每个数据报做独立的路由选择。

在这里首先要搞清楚什么是数据报。其实数据报就是在数据前部增加了源地址和目的地址信息字段的报文分段，可以作为独立的数据进行传输。因为每个数据报自身携带有足够的信息，其中包括源地址、目的地址、结点间的路由信息等，数据报的发送就是通过这些地址和路由信息来保证数据准确发送到目的节点的。一个结点接收到一个数据报后，根据数据报中的地址信息和结点所存储的路由信息，找出一个合适的出路，把数据报原样发送到下一个结点。

数据报操作方式的数据发送原理如下：

- 1) 当某个端系统要发送报文时，先将报文拆成若干个带有序号和地址信息的数据报，然后依次发给网络结点。
- 2) 各结点可根据数据报中所包括的地址和路由信息，选择不同的路由路径进行发送。另外，各个结点也可能随时根据网络的流量、故

障等情况选择最佳路径。

在这种交换方式中，各数据报各行其道，很难保证全部按顺序到达目的节点，有些数据报甚至还可能在途中丢失。图7-9所示的就是不同数据报沿不同路径传输的示例。从主机A要发送到相同目的节点——主机B的两个数据报D1、D2沿着不同路径进行传输。当然，只是可以这么做，并不是说数据报分组交换中每个数据报必须走不同的路径。

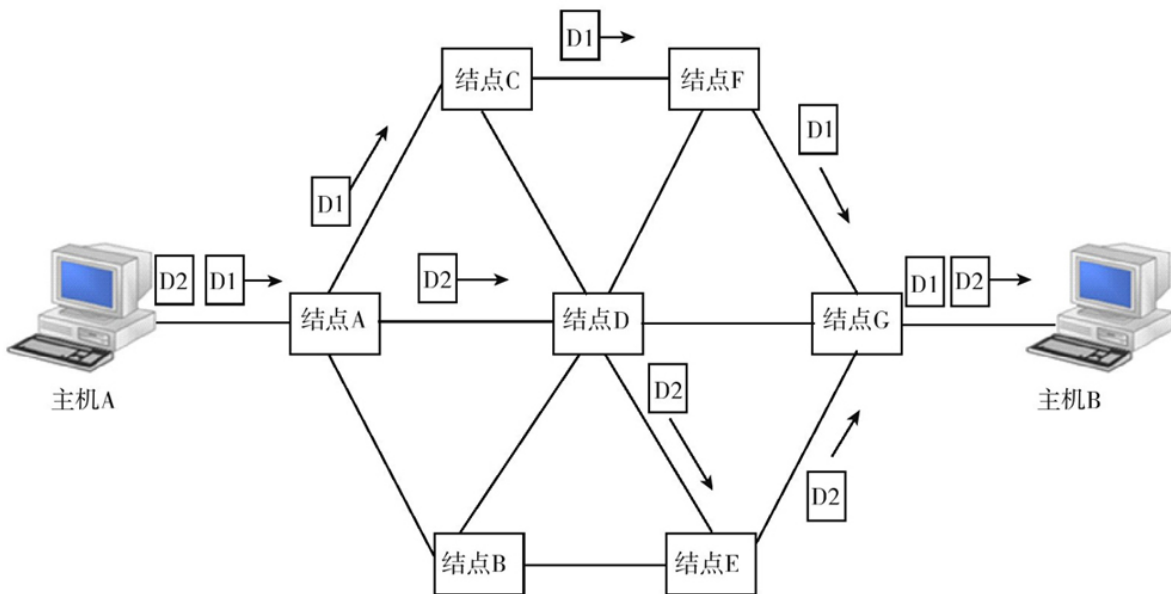


图 7-9 不同数据报沿不同路径进行交换的示例

在数据报分组交换方式中，分组传送之前不需要预先在源主机与目的主机之间建立线路连接，只需根据节点通信主机（如路由器、三层交换机）所创建的路由信息为不同的数据报选择相同或不同的传输路径。也就是说源主机所发送的每一个数据报分组都可以独立地选择

一条传输路径，同一源主机发送的不同分组在通信子网中可能通过不同的传输路径，不按顺序到达同一个目的主机。

因为数据报分组交换无须为每次通信建立单独的逻辑连接，所以数据交换效率较高，特别适用于突发性通信。但由于每一个分组在传输过程中都必须带有目的地址与源地址，所以有效数据的传输率比较低；另外，因为数据报分组每经过一个结点都要根据路由算法选择最佳路由路径，所以数据报方式报文传输延迟较大，不适用于长报文、会话式通信。

7.2.5 虚电路交换和数据报交换的比较

为了正确理解虚电路交换和数据报交换这两种不同的分组交换方式，本节将从各方面对它们进行比较。

(1) 传输方式上的区别

虚电路交换方式需在源、目的主机之间正式进行数据交换前建立逻辑连接；通信完成后，又需要释放对应的逻辑连接。而在数据报交换方式中，无须先建立连接（当然也无需释放连接），直接根据路由信息进行数据交换。

(2) 数据格式的区别

虚电路交换方式仅在源主机发出呼叫分组中需要填上源和目的主机的IP地址，在数据传输阶段，各分组都只需填上虚电路号（也就是VC号）和分组号。而数据报交换方式，由于每个数据报都单独传送，因此在每个数据报中都必须添加源和目的主机的IP地址，以便网络结点根据对应的路由信息向目的主机转发，这在频繁的人-机交互通信应用中不是很适用，也降低了信道的利用率。

(3) 转发路径的区别

虚电路交换方式同一次通信的每个分组都会沿着同样的路径进行转发，也无须进行路径选择。在数据报交换中，每个数据报每经过一个网络结点都要进行一次路径选择，而且同一报文所分出的多个报文分组所采用的转发路径也可能不一样。

(4) 可靠性方面的区别

在虚电路交换方式中，由于从源主机发出的所有分组都是通过事先建立好的一条虚电路进行传输的，所以能保证分组按发送顺序到达目的主机，且目的主机每收到一个分组后需要向源主机应答确认，可靠性高。但在数据报交换方式中，当需要把一份长报文分成若干个短的数据报分组时，由于它们都是被独立传送的，还可以各自通过不同的路径到达目的主机，因此不能保证这些数据报分组按序列到达目的主机。再加上这种交换方式的目的节点在收到数据报后也不需发送确认，所以可靠性较低。

(5) 适应性方面的区别

在虚电路交换方式中，当传输途中的某个结点或链路发生故障时，必须重新建立虚电路才能进行通信。而在数据报交换方式中，可以绕开这些故障地区另选其他路由路径，快速地把数据传至目的地。因此数据报交换比虚电路交换的适应性更强。

(6) 拥塞控制能力方面的区别

在数据报交换方式中，中间结点可为数据报选择一条流量较小的路由路径，避开流量较高的路径，因此数据报交换方式既可以平衡网络中的信息流量，又可使数据报得以更迅速地传输。而在虚电路交换方式中，一旦虚电路建立后，中间结点是不能根据流量情况来改变分组的传送路径的，所以拥塞控制能力较差。

综上所述，虚电路交换方式和数据报交换方式各有优、缺点，综合比较如表7-1所示。

表 7-1 数据报和虚电路交换方式比较

比较选项	数据报	虚电路
建立连接	不需要	需要
添加地址信息	每个分组都有源和目的 IP 地址	无需 IP 地址信息，但需加上虚电路标识符
路由选择	对每个分组独立进行，不同分组的路由路径可能不同	当虚电路建好时，路由就已确定，所有分组都经过此路由进行交换
结点失效的影响	除了在崩溃时丢失分组外，无其他影响	所有经过失效路由器的此虚电路都要被终止
数据传输效率	低	高
拥塞控制	难	如果有足够的缓冲区分配给已经建立的每条虚电路，则容易控制
可靠性	低	高
适应性	强	弱
适用环境	大多数普通数据通信	人机交互比较频繁，可靠性要求高的应用环境，如语音通信

7.3 网络层协议及报文格式

在TCP/IP体系结构的网际互连层，最重要的协议就是IP协议簇。目前主流的IPv4协议簇中包括了三个协议：IP（Internet Protocol，因特网协议）、ARP（Address Resolution Protocol，地址解析协议）、ICMP（Internet Control Message Protocol，因特网控制消息协议）。IPv6协议簇中包括了四个协议：IPv6、ICMPv6、ND（Neighbor Discovery，邻居发现）协议和MLD（Multicast Listener Discover，组播侦听器发现）协议。本节将分别对这些协议及它们的报文格式进行具体介绍。

7.3.1 IP协议基本功能

目前的计算机网络，特别是TCP/IP网络，使用最多的是数据报分组交换方式，而IP协议是用于将多个分组交换网络连接起来的最典型通信协议。IP协议是一个无连接的服务，负责在源地址和目的地址之间传送数据报，然后为了适应不同网络对分组大小的要求，需要对上层传来的报文进行分割，最后调用本地网络协议将数据报传送给下一个网关或目的计算机。经过十几年的发展，目前最新的IP协议的版本为IPv6，不过现在主流使用的仍是IPv4。

IP协议的主要功能就是把数据报在互连的网络上传送，将数据报在一个个IP模块间传送直到传送到目的模块。网络中每个主机和网关上都有IP模块。数据报在一个个模块间通过路由处理网络地址传送到目的地址。具体来说，IP协议主要有以下几方面的功能：

（1）寻址

通过第6章的学习我们已经知道，在同一以太网内部，结点间的寻址可以通过二层MAC地址进行，但在不同网络之间，是不能通过MAC地址的，因为用于MAC地址寻址的广播帧只能在同一个以太网段内部进行，不能在不同网段内传播。在不同网络中只能通过三层地址进行寻址，就像在一个国家范围内不能通过人的名字来查找一个人，而必须通过身份证来查找一样。因为同名的人可能非常多，但每个人的身份证是唯一的。当然对于不同协议的网络，三层协议也不一样，如在常用的IP网络中运行的三层协议就是IP协议，对应的三层地址就是IP地址，而在以前的Netware IPX/SPX（Internetwork Packet eXchange/Sequences Packet eXchange，网际包交换/顺序包交换）网络中三层协议就是IPX地址。

有关IPv4和IPv6地址方面的知识将在第8章具体介绍。

（2）数据报的封装

在IP网络中，从传输层到达的数据段都需要经过IP协议进行重新封装的。因为IP协议是无连接的服务，并且采用数据报交换方式，所以封装后形成的是IP数据报。IP封装的目的就是标识此IP数据报发送节点和接收节点的IP地址及控制信息。有关IP数据报的格式将在本章后面具体介绍。

（3）分段与重组

不同网络上的链路可以传输的最大报文大小是不同的，这就是我们通常所说的MTU（最大传输单元）。为了使我们要传输的数据报能在不同网络中传输，当一些尺寸较大的数据报要在某个MTU值比较小的网络链路上传输时就可能需要对原来的数据报进行拆分，形成一个个小的分段，然后再把这些分段依次传输出去。这就是IP协议的分段功能。既然在发送节点对原来的数据报进行了拆分，在接收节点自然就面临了一个如何把这些被拆分的分段重新组合起来，还原成原来的大的数据报的问题，这就是IP协议的分段重组功能。

7.3.2 IPv4的不足

IPv4自1981年发布RFC791后就没有什么更改。虽然IPv4被无数实践证明是可靠、易于实现和可交互的，并且经受住了各种考验，但时至今日随着IP网络的高速发展，当初设计的IPv4仍面临了以下这些无法解决的实际问题。

(1) IPv4地址空间面临枯竭

我们知道，IPv4地址分私网地址和公网地址两大类，私网地址就是仅用于局域网内部使用的，不同企业都可以重复使用，而公网地址则是全球唯一的，是需要注册、购买的，一个地址只能在一个地方使用。由此可见，公网IP地址可以说是用一个少一个。但IPv4地址仅32位，总体可提供的地址数有限，更别说公网地址了。这是IPv4最终走向死亡的最主要的因素。

IPv4地址已经变得相当缺乏，时至今日全球几乎都无IP地址可分了，这大大影响了计算机网络的发展，也严重不适应当前移动互联网、物联网等新兴互联网技术的发展和普及。尽管我们一直在采用像NAT（Network Address Translation，网络地址转换）、VLSM

（Variable Length Subnet Mask，可变长子网掩码）、CIDR（Classless Inter-Domain Routing，无类域间路由）这类可以更加充分使用现有公

网IP地址的技术，但仍然很难满足从企业到个人的新兴互联网应用需求，于是采用新的IPv6协议的呼声在全球范围内日益高涨。有关NAT、VLSM和CIDR等技术将在第8章中具体介绍。

（2）骨干路由器维护的路由表表项数量过多

由于IPv4发展初期的分配规划的问题，造成许多IPv4地址块分配不连续，不能有效聚合路由。针对这一问题，虽然通过采用CIDR，以及回收并再分配公网IPv4地址的方法有效抑制了全球IPv4 BGP路由表的线性增长。但目前全球IPv4 BGP路由表仍在不断增长，已经达到17万多条，经过CIDR聚合以后的BGP也将近10万条。日益庞大的路由表耗用内存较多，对设备成本和转发效率都有一定的影响，这一问题促使设备制造商不断升级其路由器产品，提高其路由寻址和转发的性能。

（3）地址结构不合理

IPv4地址的A、B、C、D和E这五种本身的分类结构就存在严重缺陷，不利于地址资源的充分利用。如果一个组织分配了A类地址，大部分的地址空间被浪费了；如果一个组织分配了C类地址，地址空间又严重不足，而且D类和E类地址都无法利用。虽然可能通过VLSM和CIDR技术来弥补，但这同样会使路由表和路由策略变得十分复杂，大大影响了路由效率。

(4) 配置复杂

目前大多数IPv4的实现方案必须手动配置，或通过控制状态的地址配置协议进行配置，例如通过动态主机配置协议（DHCP）进行配置。随着使用IP的计算机和设备越来越多，越来越需要更简单更自动化的地址配置和其他不依赖于DHCP基础结构的管理配置设置。IPv6协议就有这种自动配置功能。

(5) 服务质量差

因为IPv4是一个无连接协议，所以它本身就具有不可靠性。虽然IPv4有服务质量（QoS）标准，但是实时通信支持依赖于IPv4服务类型（ToS）字段和负载的标识，通常使用UDP或TCP端口。不幸的是，IPv4协议的ToS字段功能有限，并且有不同的解释。另外，当IPv4数据报负载被加密后，负载标识使用TCP和UDP端口是不可能的。

(6) 不支持端到端安全

因为IPsec只是IPv4中的一个可选项，且在实际部署中多数节点都不支持IPSec，所以IPv4的安全性比较差。另外，IPv4中的NAT需要对IP报文头进行修改，有时甚至需要修改相关应用数据，而在端到端安全中，IP报头的完整性通过加密来保证，报文的发出者负责保护报文头的完整性，在接收端检查收到报文的完整性，在转发过程中任何对

报头的修改都会破坏完整性检查，所以在部署NAT的情况下无法支持端到端的安全。

7.3.3 IPv6的主要优势

为了解决以上IPv4的这些不足，Internet工程任务组（IETF）又开发了一组协议和标准，那就是IPv6协议簇，其中最主要的协议当然就是IPv6了。IPv6与IPv4协议一样，也是一种无连接的、不可靠的数据报协议，主要用于在主机之间寻址和路由数据报。

新的IPv6协议相对目前正在使用的IPv4协议来说主要具有以下几方面的新特性。

（1）更大的地址空间

IPv6的地址结构中有128位（16字节），比起IPv4的32位（4字节）来说扩展了4倍，可以极大地满足未来新兴互联网技术和应用发展的地址需求。初步估算，全球每个人都可以分到差不多10个专用IP地址，到那时我们每个人都可以在互联网上有唯一的身份了。

（2）有效和分级的寻址及路由基础结构

IPv6地址长度为128位，可提供远大于IPv4的地址空间和网络前缀，因此可以方便地进行网络的层次化部署。同一组织机构在其网络中可以只使用一个前缀。对于ISP，则可获得更大的地址空间。这样ISP可以把所有客户聚合形成一个前缀并发布出去。分层聚合使全局路

由表项数量很少，转发效率更高。另外，由于地址空间巨大，同一客户使用多个ISP接入时可以同时使用不同的前缀，这样不会对全局路由表的聚合造成影响。

(3) 支持无状态和控制状态的地址配置

为了简化主机配置，IPv6既支持控制状态的地址配置，例如DHCP服务器存在时的地址配置，也支持无状态的地址配置，即DHCP服务器不存在时的地址配置。使用无状态的地址配置，链接上的主机将使用用于该链接的IPv6地址（本地链接地址）和由本地路由器广告的前缀导出的地址，自动配置它们自己。即使没有路由器，同一个链接上的主机也可以自动用本地链接地址配置自己，并在没有手动配置的情况下通信。

(4) 对服务质量（QoS）的更好支持

IPv6报头中的新字段定义了如何处理和识别通信。通过在IPv6报头中使用流标签字段，通信标识允许路由器识别属于流的数据报，并为之提供特殊处理。因为在IPv6报头中标识通信，所以即使在数据报有效负载使用IPSec加密时也很容易实现对QoS的支持。

(5) 提供用于邻近结点交互的新协议

IPv6的ND（邻居发现）协议是一个用于IPv6的Internet控制消息协议，用于管理邻居结点（即同一链接上的结点）的交互。ND协议使用有效的多播和单播消息代替原来IPv4协议中的ARP和ICMPv4，并提供附加的功能。

（6）可扩展性

IPv4中虽然也支持IPSec，但只是通过选项支持，实际部署中多数结点都不支持。但在IPv6中，IPSec是IPv6协议基本定义中的一部分，部署的任何结点都必须支持。因此，在IPv6中支持端到端安全要容易得多。IPv6中支持为IP定义的安全目标：保密性（只有预期接收者能读数据）、完整性（数据在传输过程中没有被篡改）、验证性（发送数据的实体和所宣称的实体完全一致）。

（7）支持移动特性

IPv6协议规定必须支持移动特性，任何IPv6结点都可以使用移动IP功能。与移动IPv4相比，移动IPv6使用邻居发现功能可直接实现外地网络的发现并得到转交地址，而不必使用外地代理。同时，利用路由扩展头和目的地址扩展头可实现移动结点对等结点之间的直接通信，解决了移动IPv4的三角路由、源地址过滤问题，移动通信处理效率更高且对应用层透明。

7.3.4 IPv4数据报头部格式

发送端的网络层在收到它的上一层——传输层发来的数据段时，需要通过网络层协议将其封装成数据报，也就是加上网络层IP协议（在此仅以IP网络为例进行介绍）头部。IP协议头部主要是源和目的网络的IP地址，以便可以把数据分段传输到目的网络中。然后数据报向下传输，到了数据链路层后又要封装成数据帧。

与在第4章介绍的数据帧格式中包括帧头和数据部分类似，一个IP数据报也包括报头和数据这两个组成部分，如图7-10所示。其中数据部分就是来自传输层的完整数据段，而报头部分是为了正确传输数据报而增加的网络层IPv4（此处仅以IPv4网络为例进行介绍）协议信息。



图 7-10 IPv4数据报基本结构

1.IPv4数据报头部格式

IPv4数据报头部格式如图7-11所示，下面是各个字段的说明。

4	4	8	3	13	位
版本	头部长度	区分服务	总长度		
标识			标志	段偏移	
生存时间		协议	校验和		
源地址					
目的地址					
选项				填充	

图 7-11 IPv4数据报头部格式

(1) 版本 (Version)

版本字段指定IP数据报中使用的IP协议版本，占4位。此处是IPv4版本，值为4（0100）。

(2) 头部长度 (Header Length)

头部长度字段指示IP数据报头部的总长度，IP数据报头部的总长度以4字节为单位，该字段占4位。当报头中无选项字段时，报头的总长度为5，即20字节（此时，报头长度值为0101）。这就是说IP数据报头部固定部分的长度为20字节。当IP报头长度为1111时（即十进制的

15)，头部固定部分长度就达到60字节。但报头长度必须是32位（4字节）的整数倍，如果不是，需在选项字段的填充（PAD）子字段中补0凑齐。

（3）区分服务（Differentiated Services）

最开始IP数据报的这个字段为优先级和服务类型字段，又称服务类型（ToS）字段，用于表示数据报的优先级和服务类型，占8位。它包括一个3位长度的优先级、4位长度的标志位。标志位分别是D

（Delay，延迟）、T（Throughput，吞吐量）、R（Reliability，可靠性）和C（Cost，开销），分别表示包延迟、吞吐量、可靠性和开销值，用来获得更好的服务。最高1位未用。

1998年IETF在RFC2474中把IP数据报中ToS字段改名为区分服务字段，同样为8位，前6位构成DSCP（Differentiated Services Code Point，区分服务码点），是IP优先级和服务类型字段的组合，定义了0~63共64个优先级。最后2位未使用。无论是哪种版本，该字段只有在使用区分服务时才起作用，如果没有使用区分服务，则该字段值为0。

（4）总长度（Total Length）

总长度字段标识整个IP数据报的总长度，包括报头和数据部分，整个IP数据报总长度以字节为单位，该字段占16位。由此可得出，IPv4

数据报的最大长度为 $2^{16} - 1$ 字节即65535字节（即64KB）。之所以要减1，是因为 2^{16} 中包括0这个值。

说明 在网络层下面的每一种数据链路层都有自己的帧格式，其中包括表示数据字段的最大长度，这称为最大传送单元（**Maximum Transfer Unit, MTU**）。当一个数据报封装成链路层的帧时，此数据报的总长度（包括头部和数据两部分）一定不能超过下面的数据链路层的MTU值。具体将在下节介绍IP数据报封装时介绍。

（5）标识（**Identification**）

标识字段用于表示IP数据报的标识符，占16位，每个IP数据报有一个唯一的标识。IP软件在存储器中维持一个计数器，每产生一个数据报，计数器就加1，并将此值赋给这个标识字段。但这个标识并不是序号，因为IP是无连接服务，数据报不存在按序接收的问题。当数据报由于长度超过下面数据链路层的MTU值而必须分段时，这个标识字段的值就被复制到所有的数据报分段的标识字段中。相同的标识字段的值使分段后的各数据报分段最后能正确地重装成为原来的数据报。

（6）标志（**Flags**）

标志字段指出该IP数据报后面是否还有分段，也就是这个字段是分段标志，占3位。目前只有前两位有意义：最低1位记为MF（**More Fragment**），如果MF=1，即表示后面还有分段，如果MF=0表示这已是

某个数据报的最后一个分段；中间1位记为DF（Don't Fragment），当DF=1时表示不允许分段，DF=0时表示允许分段；最高1位没有使用。

（7）段偏移（Fragment Offset）

段偏移字段用以指出该分段在数据报中的相对位置，也就是说，相对于用户数据字段的起点，该分段从何处开始，占13位。若有分段，段偏移以8字节为偏移单位，即每个分段的长度一定是8字节（64位）的整数倍。第一个分段的段偏移值就是000000000000000，如果第一个分段一共是64字节，则第二个分段的偏移值为0000000001001，相当于十进制数的9，因为它是从第9个“8字节”数据块开始的。如果没有分段，则该字段值为0。

（8）生存时间（Time To Live, TTL）

生存时间字段用来标识IP数据报在网络中传输的有效期，以秒来计数，占8位。最初的设计是以秒作为TTL的单位，每经过一个路由器时，就在TTL中减去数据报在路由器消耗掉的一段时间。若数据报在路由器消耗的时间小于1s，就把TTL值减1。TTL的建议值是32s，最长为 $2^8 - 1 = 255s$ 。现在通常认为这个TTL是指数据报允许经过的路由器数，每经过一个路由器，则TTL减1，当TTL值为0时，就丢弃这个数据报。设定生存时间是为了防止数据报在网络中无限制地循环转发。

（9）协议（Protocol）

协议字段用来标识此IP数据报在传输层所采用的协议类型（如TCP、UDP或ICMP等），以便使目的主机的IP层知道应将数据部分上交给哪个处理过程，占8位。如TCP的协议号是6，等于二进制的00000110，UDP的协议号是17，等于二进制的00010001。

（10）校验和（checksum）

校验和字段用来检验IP数据报的报头部分（不包括“数据”部分）在传输到接收端后是否发生了变化，占16位。这是因为数据报每经过一个路由器，路由器都要重新计算一下报头检验和（因为一些字段，如生存时间、标志、段偏移等都可能发生变化），不检验数据部分可减少计算的工作量。

经验之谈 利用校验和字段检验报头部分数据正确性的基本原理是：先在发送端校验和字段中填上一个特定的值，然后在接收端把包括校验和字段在内的报头部分进行二进制反码求和，再取反，如果结果为0，则表示报头部分在传输过程中没有发生变化，否则表示在传输过程中出现了差错。从以上可以看出，这里最关键的是在发送端计算出这个校验和的值。它的计算步骤如下：

- 1) 把IP数据报报头中的校验和字段置0。
- 2) 把头部看成由以16位（2字节）为单位的数字组成，对每16位的二进制反码进行求和。如报头长度不是16位的整数倍数，则用0填充

到16位的整数倍数。若此时校验和字段值为0，可以不计，因为0的反码仍为0。

3) 以上得到的结果就是我们要求的校验和字段值，系统自动将其填入IP数据报报头的校验和字段中。

在接收端，同样按照以16位为单位，对IP数据报报头部分进行二进制反码求和，再取反，如果结果为0，表示报头部分在传输过程中没有发生改变，否则表示发生了差错。但要注意，此时因为校验和字段已不再是0了，而是等于除了校验和字段外其他字段的反码之和。现在再对校验和字段值取反求和，再与其他字段的反码之和（相当于原来“校验和”字段的值）相加，结果肯定是全为1，因为这两个值互为反码；再取反后，结果肯定为0。这就是校验和的基本原理。

举个简单的例子，假设有3个数（为了简便，在此均用4位表示）：2（0010）、3（0011）、C（代表校验和字段值），计算C，即求2和3的反码之和，得到9（1001）。现在假设把这3个数（2，3和C）传送到接收端。在接收端也要对这3个数进行反码求和。因为2和3这两个的反码之和我们在计算C时就已得出了，就是9（1001），现在只需对C（“校验和”字段值）进行求反，得到6（0110）。把1001与0110相加，得到15（1111）。再取反，就得到了0。这就是这3个数在传输过程没有出现差错的情况下得到的，这就是校验和的校验原理。

（11）源地址/目的地址（Source Address/Destination Address）

源地址/目的地址这两个字段分别表示该IP数据报发送者和接收者的IP地址，各占32位。在整个数据报传送过程中，无论经过什么路由，无论如何分段，此两字段一直保持不变。

（12）选项（Options）

选项字段支持各种选项，提供扩展余地。根据选项的不同，该字段是可变长的，从1字节到40字节。用来支持排错、测量以及安全等措施。作为选项，用户可以使用，也可以不使用它们。但作为IP协议的组成部分，所有实现IP协议的设备都必须能处理IP选项。在使用选项的过程中，如果造成IP数据报的报头不是32位的整数倍，这时需要使用后面的填充字段凑齐。如果恰好为整数倍，则不需要填充字段。

2.IP数据报报头示例

图7-12所示的是在科来分析仪上通过抓包得出的IP数据报各字段值的示例，从中也可以全面体现图7-11所示的IP数据报头部格式。下面介绍从中发现的各字段值。

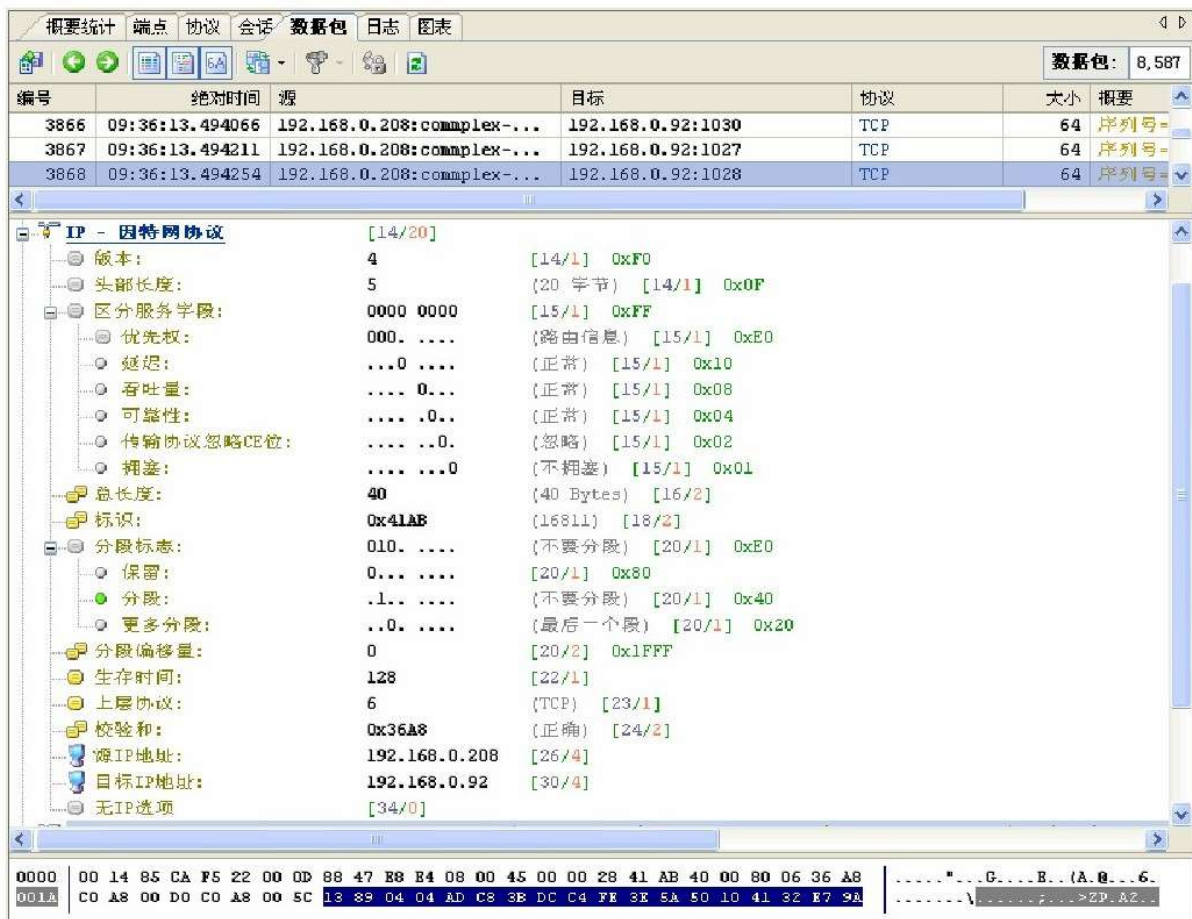


图 7-12 IP数据报头部格式示例

- 版本: 4, 表示当前网络中为IPv4。
- 头部长度: 5, 表示IP报头长度为4字节×5=20字节。
- 区分服务字段: 00000000, 表示当前IP数据报中没有使用服务类型字段。
- 总长度: 40, 表示该数据报总长为40字节。

□标识：表示该数据报的标识为**0x41AB**（十六进制）。

□分段标志：**010**，第2位为**1**，表示该数据报不能被分段，最低位为**0**，表示后面没有分段。

□分段偏移量：**0**，表示没有被分段。

□生存时间：**128**，表示该数据报最多可以经过**128**个路由。

□上层协议：**6**，表示IP数据报的传输层协议为**TCP**协议。

□校验和：该数据报校验和为**0x36A8**（正确），表示该数据报是完整的。**0x36A8**这个值是根据本节前面介绍的校验和字段计算原理得出的。

□源IP地址：**192.168.0.208**，表示发送该数据报的源节点IP地址为**192.168.0.208**。

□目标IP地址：**192.168.0.92**，表示该数据报发送的目的节点IP地址为**192.168.0.92**。

□无IP选项：表示该数据报没有选项字段。

7.3.5 IPv6数据报头部格式

RFC 2460定义了IPv6数据报格式。总体结构上，IPv6数据报格式与IPv4数据报格式是一样的，也是由IP报头和数据（在IPv6中称为有效载荷）这两部分组成，但在IPv6数据报数据部分还可以包括0个或多个IPv6扩展报头（Extension headers），如图7-13所示。IP报头部分固定为40字节长度，而有效载荷部分最长不得超过65535字节。

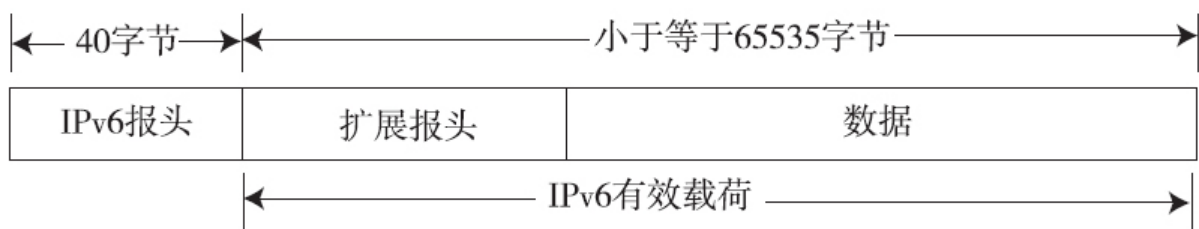


图 7-13 IPv6数据报结构

IPv6和IPv4之间最显著的区别为：IP地址的长度从32位增加到128位。

通过裁减IPv4报头中的某些字段，或把一些字段移入到扩展报头中，IPv6基本报头的最大总长度大大减小了。IPv6使用固定长度的基本报头，从而简化了转发设备对IPv6报文的处理，提高了转发效率。尽管IPv6地址长度是IPv4地址长度的4倍，但IPv6基本报头的长度只有40字节，为固定的IPv4报文头长度（不包括选项字段）的2倍。IPv6报头

格式如图7-14所示。下面是各字段介绍，关键是要理解各个字段的作用。

4	4	8	8	8	位
版本	通信分类	流标签			
有效载荷长度			下一个头部	跳数限制	
源IP地址（128位）					
目的IP地址（128位）					

图 7-14 IPv6报头格式

（1）版本（Version）

版本字段用来表示IP数据报使用的是IPv6协议封装，占4位，对应值为6（0110）。

（2）通信分类（Traffic Class）

通信分类字段用来标识对应IPv6数据报的通信流类别，或者说是优先级别，占8位，类似于IPv4数据报中的ToS（服务类型）字段。

（3）流标签（Flow Label）

流标签字段是IPv6数据报中新增的一个字段，占20位，可用来标记报文的数据流类型，以便在网络层区分不同的报文。流标签字段由源节点分配，通过流标签、源地址、目的地址三元组方式就可以唯一标识一条通信流，而不用像IPv4那样需要使用五元组方式（源地址、目的地址、源端口、目的端口和传输层协议号）。这样发动的最大好处有两点：一是流标签可以和任意的流关联，需要标识不同类型的流（可以是非五元组）时，无须对流标签做改动；二是流标签在IPv6基本头中，使用IPSec时此域对转发路由器可见，因此转发路由器可以在使用IPv6报文IPSec的情况下仍然可以通过三元组（流标签、源地址、目的地址）针对特定的流进行QoS（质量服务）处理。

（4）有效载荷长度（Payload Length）

有效载荷长度字段是以字节为单位标识IPv6数据报中有效载荷部分（包括所有扩展报头部分）的总长度，也就是除IPv6基本报头以外其他部分的总长度，占20位。

（5）下一个头部（Next Header）

下一个头部字段用来标识当前报头（或扩展报头）的下一个头部的类型，占8位。每种扩展报头都有其对应的值，具体将在下节介绍。下一个头部字段内定义的扩展报头类型与IPv4报文中的协议字段值类似，但在IPv6数据报中，紧接着IPv6报头的可能不是上层协议头部（当

没有扩展报头或者为最后一个扩展报头时后面才是上层协议头），而是IPv6扩展报头。这一机制下处理扩展头更高效，因为标识了数据报中对应的上层协议或扩展报头类型，转发路由器只处理必须处理的选项扩展报头，提高了转发效率。

（6）跳数限制（Hop Limit）

跳数限制字段与IPv4报文中的TTL字段类似，指定了报文可以有效转发的次数，占8位。报文每经过一个路由器结点，跳数值就减1，当此字段值减到0时，则直接丢弃该报文。

经验之谈 同一路由器上所连接的多个网段之间的跳数为0，也就是在同一路由器上所连接的多个网络是直接相通的，不用配置路由。这也是像网关、路由器这类设备通过它们上面的多个端口就可以连接多个不同网段的原因。但是不同网关、路由器上连接的不同网段是必须要配置路由的。

另外，之所以称为“网关”，是因为在这类设备上仅支持直接连接的多个网段（每个网络连接在一个端口上）间通过它本身即可实现的互通，不提供到达非直接连接网络（也就是在其它三层设备上连接的网络）的路由功能。但要注意的是，我们在计算机等设备上配置的网关并不是指与它所直接连接交换机端口的IP地址，而是指三层设备上连接该计算机所在网段的那个端口的IP地址。网关IP地址必须是位于三

层设备（如网关、路由器、三层交换机）端口上的，而且必须位于网络边缘（也就是该设备的其他端口必须连接的是不同的网段）的端口上。

（7）源IP地址（Source IP Address）

源IP地址字段标识了发送该IPv6报文源节点的IPv6地址，占128位。

（8）目的IP地址（Destination IP Address）

目的IP地址字段标识了IPv6报文的接收节点的IPv6地址，占128位。

对比图7-11中的IPv4数据报头部格式可以看出，IPv6去除了IPv4报头中的头部长度的标识、标志、段偏移、校验和、选项、填充这么多字段，却只增了流标签这一个字段，因此IPv6报头处理和IPv4相比大大简化，提高了处理效率。另外，IPv6为了更好地支持各种选项处理，提出了扩展头的概念，新增选项时不必修改现有结构就能做到，理论上可以无限扩展，体现了优异的灵活性。

7.3.6 IPv6扩展报头

在图7-14中可以看到，IPv6报文中可以携带可选的IPv6扩展报头。IPv6扩展报头是跟在IPv6基本报头后面的可选报头。由于在IPv4的报头中包含了几乎所有的可选项，因此每个中间路由器都必须检查这些选项是否存在。在IPv6中，相关选项被统一移到了扩展报头中，这样中间路由器不必处理每一个可能出现的选项（仅有“逐跳选项”报头是必须要处理的），提高了路由器处理数据报文的速度，也提高了其转发性能。

IPv6扩展报头附加在IPv6报头目的IP地址字段后面，可以有0个，或者多个扩展报头。主要的IPv6扩展报头有以下几类：

（1）逐跳选项头（Hop-by-Hop Options Header）。

本扩展报头类型值为0（在IPv6报头下一个头部字段中定义，下同）。此扩展报头须被转发路径所有结点处理。目前在路由告警

（RSVP和MLDv1）与Jumbo帧处理中使用了逐跳选项头，因为路由告警需要通知到转发路径中所有结点，而Jumbo帧是长度超过65535字节的报文，传输这种报文需要转发路径中所有结点都能正常处理。

（2）目的选项头（Destination Options Header）。

本扩展报头类型值为60。只可能出现在两个位置：①路由头前，这时此选项头被目的节点和路由头中指定的结点处理；②上层头前（任何ESP头后），此时只能被目的节点处理。

移动IPv6中使用了目的选项头，称为家乡地址选项。家乡地址选项由目的选项头携带，用于移动结点离开“家乡”后通知接收节点此移动结点对应的家乡地址。接收节点收到带有家乡地址选项的报文后，会把家乡地址选项中源地址（移动节点的家乡地址）和报文中源地址（移动节点的转交地址）交换，这样上层协议始终认为是在和移动节点的家乡地址通信，实现了移动漫游功能。

（3）路由头（Routing Header）。

本扩展报头类型值为43，用于源路由选项和移动IPv6。

（4）分段头（Fragment Header）。

本扩展报头类型值为44，用于标识数据报的分段，在IPv4报头中就有对应的字段。当源节点发送的报文超过传输链路MTU（源节点和目的节点之间传输路径的MTU）时，需要对报文进行分段时使用。

（5）认证头（Authentication Header）。

本扩展报头类型值为51，用于IPSec，提供报文验证、完整性检查。

（6）封装安全有效载荷头（Encapsulating Security Payload Header）。

本扩展报头类型值为50，用于IPSec，提供报文验证、完整性检查和加密。

（7）上层头

这是用来标识数据报中上层协议类型，如TCP、UDP、ICMP等。

注意 目的选项头最多出现两次：一次在路由头前，一次在上层协议头前，其他选项头最多出现一次。IPv6节点必须能够处理选项头（逐跳选项头除外，它固定只能紧随基本报头之后）在任意位置出现，以保证互通性。

7.3.7 IPv4数据报的封装与解封装

因为我们目前使用的主要还是IPv4协议，所以重点还是介绍与IPv4相关的知识。本节要介绍IPv4数据报的封装和解封装。为了简便，IPv4数据报仍简称IP数据报，除非特别说明，本章后面的所有IP数据报就是指IPv4数据报。

发送端网络层生成的IP数据报还要继续向下传输，到达数据链路层后就要封装成数据帧了。IP数据报的“帧封装”原理很简单，只需把来自网络层的整个IP数据报（包括包头部分和数据部分）当作数据链路层帧的数据（data）部分，然后在前面加上与数据链路层对应的协议头即可。在以太网局域网中，由于数据链路层分成了LLC子层和MAC子层这两层，所以来自网络层的IP数据报在到达数据链路层后先要经过LLC子层和MAC子层的协议头封装，最终形成数据链路层的以太网MAC帧，如图7-15所示。在帧头中我们一般要加上了源和目的节点的MAC地址，因为数据链路层是通过MAC地址进行寻址的，具体参见6.4.2节。

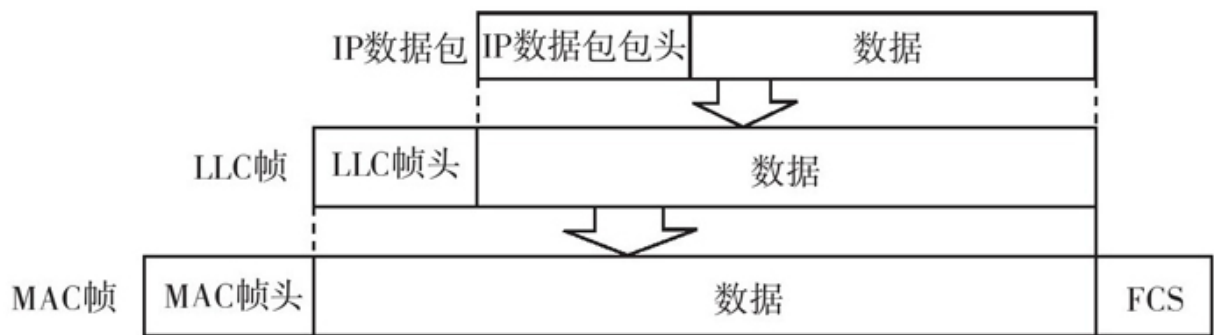


图 7-15 以太网中的IP数据报封装流程

以上所述是经过一个网络的情形，如果一个IP数据报在整个传输过程中要经过几个网络时，怎么办呢？这时其实又涉及一个解封装的过程。解封装就是由数据链路层的帧格式解封成原始的数据报格式。

图7-16所示是源主机发送一个IP数据报经过两个路由器连接的三个网络时的封装和解封装过程。下面是具体的流程。

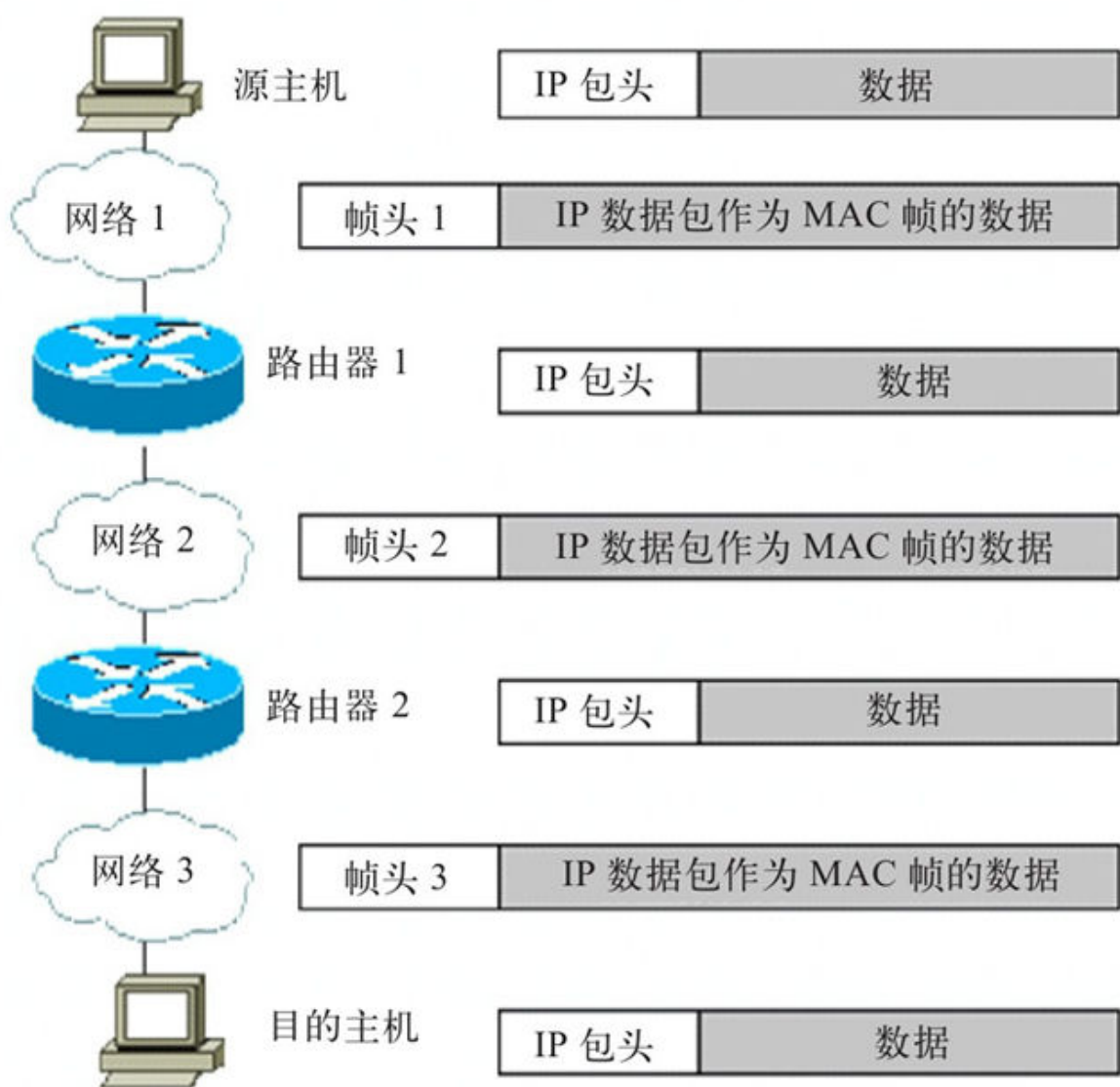


图 7-16 IP数据报经过多个网络时的封装和解封装流程

1) 当源主机发送一个IP数据报时，在内网网络1中传输是以帧形式进行的，所以首先需要把原始的IP数据报封装成帧，在网络1的链路上进行传输。此时把原来IP数据报作为整个MAC帧的“数据”部分，然后加上网络1链路层协议的头部信息，作为帧头，即帧头1。

2) 当帧传输到路由器1连接网络1的接口时，相当于要把帧从数据链路层上传到网络层，于是要对原来的帧进行解封装，去掉帧头和帧尾信息（在有帧尾的情况下），还原出原来的IP数据报，以识别包中的目的地址信息，然后根据路由器中的路由表信息进行路由转发。此时IP数据报的内容都没变，包括IP包头部分中的源和目的地址信息等。

3) 当IP数据报在路由器1中从网络1路由到网络2后，又需要在网络2的链路层进行传输，所以又要重新封装成帧，仍把原来的整个IP数据报作为数据部分，不过此时加上的是网络2链路层的协议头信息作为帧头部，即帧头2。

4) 当帧传输到路由器2连接网络2的接口时，又相当于从数据链路层到了网络层，所以又要对帧进行解封装，仍旧还原出原来的IP数据报，使路由器2可以识别包中的目的地址信息，只有这样路由器2才可以根据其路由表信息进行正确的数据报路由、转发。

5) 当IP数据报在路由器2中从网络2路由到网络3后，又需要在网络3的链路层进行传输，所以又要重新封装成帧，仍把原来的整个IP数据报作为数据部分，不过此时加上的是网络3链路层的协议头信息作为帧头部，即帧头3。

6) 当从网络3的数据链路层把帧传输到目的主机时，在目的主机上又会对帧进行解封装，去掉帧头和帧尾，还原出原来的IP数据报，

以获取IP数据报中的真正数据。

从以上过程可以看出，IP数据报无论经过了多少个网络，整个数据报内容都是不会改变的，包括包头部分的源和目的地址信息。变化的只是在不同网络数据链路上传输的帧头信息，即在不同网络链路上传输的帧源MAC地址会改变的，目的MAC地址不会改变。

7.3.8 IPv4数据报的分段与重组

在网络层中还涉及一个分段的问题，那是因为不同网络线路上可以传输的数据报大小是有限制的，且可能是不同的，也就是我们通常所说的MTU（最大传输单元），所以如果一个网络中收到的数据报太大，不能在目的网络中一次传输的话，就要对原来的包进行拆分，分成一个个小的数据报再进行传输，这就是本节将要介绍的IP数据报分段。在发送端进行了分段，在接收端自然就要把发送来的一个个分段重新按顺序组合，恢复成原始的大数据报，这就是本节要介绍的另一个问题——IP数据报重组。

当一个IPv4数据报封装成数据链路层的帧时，此数据报的总长度（即包头加上数据部分）一定不能超过表7-2所列的不同网络的数据链路层的MTU值。因此，一个IP数据报的长度只有小于或等于一个网络的MTU时才能在这个网络中进行传输。当原来的IP数据报长度大于对应网络链路的MTU时怎么办呢？这时就得对这个大的IP数据报进行拆分，分成多个符合对应网络链路MTU要求的小的数据分段。

表 7-2 不同网络数据链路层协议的 MTU 值

协议	MTU/ 字节	协议	MTU/ 字节
令牌环（16Mbps）	17914	以太网	1500
令牌环（4Mbps）	4464	X.25	576
FDDI	4352	PPP	296

在互联网中，包含了各种各样的异构网络，一个路由器也可能连接着具有不同MTU值的多个网络，从一个网络上接收的IP数据报并不一定能在另一个网络上发送该数据报。如图7-17所示，一个路由器连接两个网络，其中一个网络为以太网，根据表7-2可知其MTU值为1500字节，另一个为X.25网络，根据表7-2可知其MTU值为576字节。

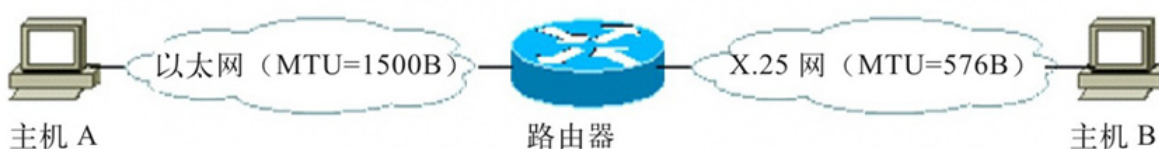


图 7-17 路由器连接不同MTU值网络的IP数据报传输示例

在本示例中，主机A连接着MTU值为1500字节的以太网络，这样在以太网上链路层传输的帧中数据部分，也就是IP数据报长度不得超过1500字节，而主机B连接着MTU值为576字节的X.25网络，这样在X.25网络链路传输的帧中的数据部分，也就是IP数据报长度不得超过576字节，即两个网络中链路上传输的MTU值是不一样的。这时如果主机A要把一个1450字节的数据报发送给主机B，路由器可以接收到主机A发送的数据报，但却不能在X.25网络上转发它，因为它超过了X.25网络链路上MTU的限制。

为了解决这个问题，就需要在路由器上对接收到的主机A发来的IP数据报进行分段，然后再将每个分段独立地进行发送，可以像正常的IP数据报一样经过独立的路由选择等处理过程，最终到达目的主机。

最后，还需要在目的主机上对这些分段进行重组，恢复原始的大IP数据报。

这时就涉及一个问题，IP数据报分段后是否在每个分段中加上对应的IP包头信息。答案是肯定的。因为分段后的每个IP包分段都需要进行单独的路由，所以必须要有相应的IP包头信息。这时就得把原来IP数据报中的包头部分复制到所有由同一个IP数据报拆分的分段上。另外，还需要在每个分段的包头部分对标志字段和段偏移字段值进行对应的修改。

如果是第一个分段，则标志字段的第一位（最低位）置1（表示后面还有分段），中间一位置0（表示允许分段），段偏移字段值为0000000000000000；如果是中间分段，则标志字段的第一位（最低位）置1，中间一位置0，段偏移字段值为相应的偏移值（以8个字节为单位）；如果是最后一个分段，则“标志”字段的第一位（最低位）置0，中间一位置0（表示此分段是最后一个分段），段偏移字段值也为相应的偏移值。包头的其他部分与原始IP数据报一样。

在路由器或主机上对IP数据报进行了拆分后，在最终的目的主机上要将接收到的所有分段进行重新组装，这就是IP数据报分段的重组过程。IP数据分段的重组是根据数据报的标识符（由同一个IP数据报拆分的各分段标识符字段是相同的）、段偏移、标志等字段进行的，按照原来拆分的顺序拼接起来，但拼接时只保留第一个分段的包头信

息，分段的包头被去掉，同时修改第一个分段包头信息，不设置标志字段和段偏移字段。

7.3.9 ARP协议报文格式及ARP表

ARP（Address Resolution Protocol，地址解析协议）是将IP地址解析为以太网MAC地址（或称物理地址）的协议。在局域网中，当主机或其他网络设备有数据要发送给另一个主机或设备时，它必须知道对方的网络层地址（即IP地址）。但是仅仅有IP地址是不够的，因为IP数据报文必须封装成帧才能通过物理网络发送，因此发送站还必须有接收站的物理地址，所以需要有一个从IP地址到物理地址的映射。APR就是实现这个功能的协议。

1.ARP报文格式

ARP是一个独立的三层协议，所以ARP报文在向数据链路层传输时不需要经过IP协议的封装，而是直接生成自己的报文，其中包括ARP报头，到数据链路层后再由对应的数据链路层协议（如以太网协议）进行封装。ARP报文分为ARP请求和ARP应答报文两种，它们的报文格式可以统一为如图7-18所示。下面是各字段的说明。

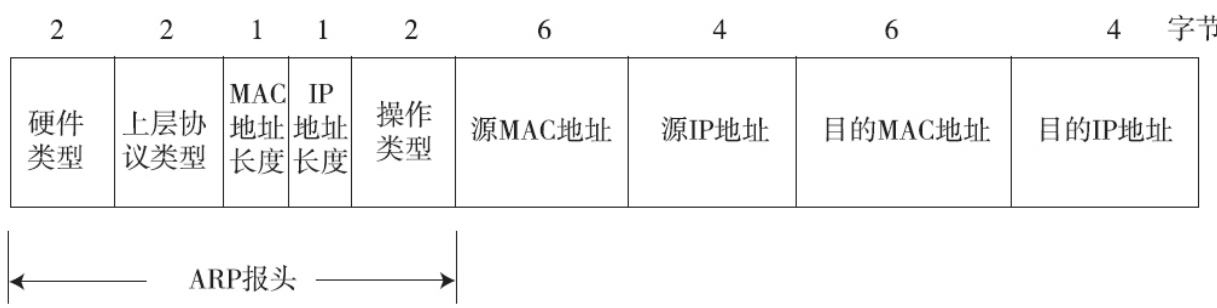


图 7-18 ARP报文格式

□硬件类型：占2字节，表示ARP报文可以在哪种类型的网络上传输，值为1时表示为以太网地址。

□上层协议类型：占2字节，表示硬件地址要映射的协议地址类型，映射IP地址时的值为0x0800。

□MAC地址长度：占1字节，标识MAC地址长度，以字节为单位，此处为6。

□IP协议地址长度：占1字节，标识IP地址长度，以字节为单位，此处为4。

□操作类型：占2字节，指定本次ARP报文类型。1表示ARP请求报文，2表示ARP应答报文。

□源MAC地址：占6字节，表示发送方设备的硬件地址。

□源IP地址：占4字节，表示发送方设备的IP地址。

□目的MAC地址：占6字节，表示接收方设备的硬件地址，在请求报文中该字段值全为0，即00-00-00-00-00-00，表示任意地址，因为现在不知道这个MAC地址。

□目的IP地址：占4字节，表示接收方设备的IP地址。

ARP报文并不是直接在网络层上发送的，它还是需要向下传输到数据链路层，所以当ARP报文传输到数据链路后，需要再次进行封装。以以太网为例，ARP报文传输到以太网数据链路层后会形成ARP帧。ARP帧格式如图7-19所示，它就是在ARP报文前面加了一个以太网帧头。

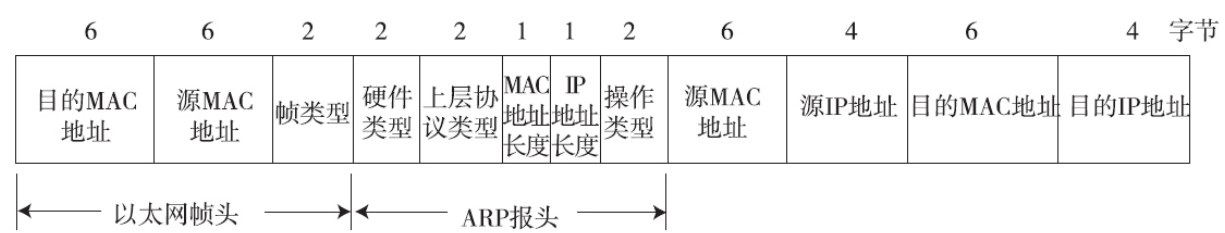


图 7-19 ARP帧格式

以太网帧头中的三个字段说明如下：

□目的MAC地址：占6字节，如果是ARP请求帧，因为它是一个广播帧，所以要填上广播MAC地址——FF-FF-FF-FF，其目标是网络上的所有主机。

□源MAC地址：占6字节，这是发送ARP帧的节点MAC地址。

□帧类型：占2字节，这里用来标识帧封装的上层协议，因为本帧的数据部分是ARP报文，所以直接用ARP的协议号0x0806表示即可。

2.ARP映射表

无论是主机，还是交换机都会有一个用来缓存同一网段设备IP地址和MAC地址的ARP映射表，用于数据帧的转发。设备通过ARP解析到目的MAC地址后，将会在自己的ARP映射表中增加IP地址到MAC地址的映射表项，以用于后续到同一目的地数据帧的转发。

ARP表项分为动态ARP表项和静态ARP表项。

（1）动态ARP表项

动态ARP表项由ARP协议通过ARP报文自动生成和维护，可以被老化，可以被新的ARP报文更新，也可以被静态ARP表项所覆盖。当到达老化时间或接口关闭时会删除相应的动态ARP表项。

（2）静态ARP表项

静态ARP表项通过手工配置（通过对应设备的IP地址与MAC地址绑定命令进行）和维护，不会被老化，也不会被动态ARP表项覆盖。配置静态ARP表项可以增加通信的安全性，因为静态ARP表项可以限制和指定IP地址的设备通信时只使用指定的MAC地址（也就是我们通常所说的IP地址与MAC地址的绑定），此时攻击报文无法修改此表项的IP地址和MAC地址的映射关系，从而保护了本设备和指定设备间的正常通信。

静态ARP表项又分为短静态ARP表项和长静态ARP表项。

□长静态ARP表项。在配置长静态ARP表项时，除了配置IP地址和MAC地址项外，还必须配置该ARP表项所对应的VLAN（虚拟局域网）和出接口。也就是长静态ARP表项同时绑定了IP地址、MAC地址、VLAN和端口，可以直接用于报文转发。

□短静态ARP表项。在配置短静态ARP表项时，只需要配置IP地址和MAC地址项。如果出接口是三层以太网接口，短静态ARP表项可以直接用于报文转发；如果出接口是VLAN虚接口，短静态ARP表项不能直接用于报文转发，当要发送IP数据包时，先发送ARP请求报文，如果收到的响应报文中的源IP地址和源MAC地址与所配置的IP地址和MAC地址相同，则将接收ARP响应报文的接口加入该静态ARP表项中，之后就可以用于IP数据包的转发了。

7.3.10 ARP地址解析原理

在图7-20的示例中，现假设主机A和B在同一个网段（均位于192.168.1.0/24网段），主机A要向主机B发送信息。主机A和主机B的IP地址和MAC地址均在图中已有标识，此时主机A已知道主机B的IP地址，要想获得主机B的MAC地址，具体的地址解析过程如下。

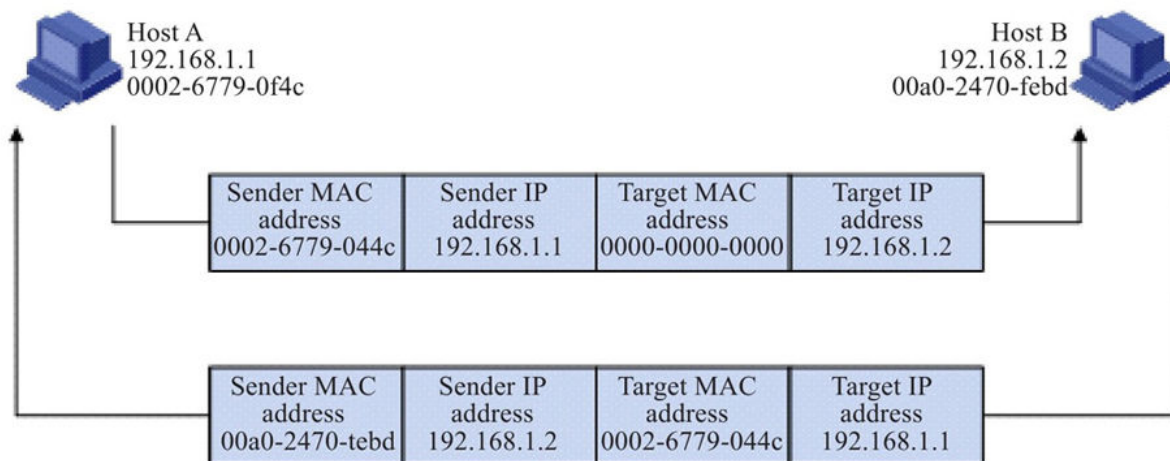


图 7-20 ARP地址解析示例

1) 主机A首先查看自己的ARP表（它是一个IP地址与MAC地址的映射表），确定其中是否包含有主机B的IP地址和对应的MAC地址。如果找到了对应的MAC地址，则主机A直接利用ARP表中的MAC地址对IP数据包进行帧封装，并将数据包发送给主机B。

2) 如果主机A在ARP表中找不到对应的MAC地址，则先缓存该数据报文，然后以广播方式（目的MAC地址为广播MAC地址——

FFFFFF，任一同网段的节点均可收到）发送一个ARP请求报文。ARP请求报文中的发送端（源）IP地址和发送端MAC地址分别为主机A的IP地址（192.168.1.1）和MAC地址（0002-6779-0f4c），目的IP地址和目的MAC地址为主机B的IP地址（192.168.1.2）和全0的MAC地址。因为ARP请求报文是以广播发方式发送，所以该网段上的所有主机都可以接收到该请求包，但只有其IP地址与目的IP地址一致的主机B会对该请求进行处理。

3) 主机B将ARP请求报文中的发送端（即主机A）的IP地址和MAC地址存入自己的ARP表中。然后以单播方式向主机A发送一个ARP响应报文，应答报文中就包含了自己的MAC地址，也就是原来在请求报文中要请求的目的MAC地址。

4) 主机A在收到来自主机B的ARP响应报文后，将主机B的MAC地址加入到自己的ARP表中以用于后续报文的转发，同时将原来缓存的IP数据包再次修改（在“目的MAC地址”字段填上主机B的MAC地址）后发送出去。

这就是同一网段中两主机的ARP地址解析的全过程。如果两主机不在同一，它们之间又是如何通信的呢？具体步骤如下：

1) 如果主机A不知道网关的MAC地址（也就是在主机A的ARP表中没有网关对应的MAC地址表项），则主机A先在本网段中发出一个

ARP请求广播，ARP请求报文中的目的IP地址为网关IP地址，代表其目的就是想获得网关的MAC地址。如果主机A已知网关的MAC地址，则略过此步。

2) 网关收到ARP广播包后同样会向主机A发回一个ARP应答包。当主机A从收到的应答报文中获得网关的MAC地址后，在主机A向主机B发送的原报文的目MAC地址字段填上网关的MAC地址后发给网关。

3) 如果网关的ARP表中已有主机B对应的MAC地址，则网关直接将在来自主机A的报文中的目MAC地址字段填上主机B的MAC地址后转发给主机B。

4) 如果网关ARP表中没有主机B的MAC地址，网关会再次向主机B所在网段发送ARP广播请求，此时目的IP地址为主机B的IP地址，当网关从收到来自主机B的应答报文中获得主机B的MAC地址后，就可以将由主机A发来的报文重新在目的MAC地址字段填上主机B的MAC地址后发给主机B。

7.3.11 ICMP协议及报文格式

ICMP是（Internet Control Message Protocol）Internet控制报文协议。它是IPv4协议簇中的一个子协议，用于在IP主机、路由器之间传递控制消息。控制消息是指网络通不通、主机是否可达、路由是否可用等网络本身的消息。这些控制消息虽然并不传输用户数据，但是对于用户数据的传递起着重要的作用。

但与ARP协议不同，ICMP依靠IP协议来完成其任务，所以ICMP报文中要封装IP头部。它与传输层协议（如TCP和UDP）的目的不同，一般并不用来在端系统之间来传送数据，不被用户网络程序直接使用，除了像Ping和tracert这样的诊断程序。

1.ICMP消息类型

ICMP报告无法传送的数据报的错误，并帮助对这些错误进行疑难解答。例如，如果IPv4不能将数据报传送到目标主机，则路由器上的或目标主机上的ICMP会向主机发送一条“无法到达目标”消息。表7-3所示为最常见的ICMP消息，并作了说明。

表 7-3 最常见的 ICMP 消息

ICMP 消息类型	用途说明
回显请求	Ping 工具通过发送 ICMP 回显消息检查特定节点的 IPv4 连接以排查网络问题。类型值为 0
回显应答	节点发送回显答复消息响应 ICMP 回显消息。类型值为 8
重定向	路由器发送“重定向”消息，告诉发送主机到目标 IPv4 地址更好的路由。类型值为 5
源抑制	路由器发送“源结束”消息，告诉发送主机它们的 IPv4 数据报将被丢弃——因为路由器上发生了拥塞。于是，发送主机将以较低的频度发送数据报。类型值为 4
超时	这个消息有两种用途。第一，当超过 IP 生存期时向发送系统发出错误信息。第二，如果分段的 IP 数据报没有在某种时限内重新组合，这个消息将通知发送系统。类型值为 11
无法到达目标	路由器和目标主机发送“无法到达目标”消息，通知发送主机它们的数据报无法传送。类型值为 3

在表7-3中的“无法到达目标”消息中还可细分，具体如表7-4所示。

表 7-4 ICMP 中常见的“无法到达目标”消息

无法到达目标消息	说 明
不能访问主机	路由器找不到到目标 IPv4 地址的路由时发送“不能访问主机”消息
无法访问协议	目标 IPv4 节点无法将 IPv4 标头中的“协议”字段与当前使用的 IPv4 客户端协议相匹配时会发送“无法访问协议”消息
无法访问端口	IPv4 节点在无法将 UDP 标头中的“目标端口”字段与使用该 UDP 端口的应用程序相匹配时发送“无法访问端口”消息
需要分段但设置了 DF	当必须分段但发送节点在 IPv4 标头中设置了“不分段”(DF)标志时，IPv4 路由器会发送“需要分段但设置了 DF”消息

ICMP协议只是试图报告错误，并对特定的情况提供反馈，但最终并没有使IPv4成为一个可靠的协议。ICMP消息是以未确认的IPv4数据报传送的，它们自己也不可靠。

2.ICMP报头格式

ICMP报文包含在IP数据报中，IP报头在ICMP报文的最前面。一个ICMP报文包括IP报头（至少20字节）、ICMP报头（至少8字节）和ICMP报文（属于ICMP报文的数据部分）。当IP报头中的协议字段值为

1时，就说明这是一个ICMP报文。ICMP报头格式如图7-21所示。各字段的说明如下。



图 7-21 ICMP报头格式

□类型：占1字节，标识ICMP报文的类型，目前已定义了14种，从类型值来看ICMP报文可分为二大类。第1类是取值为1~127的差错报文，第2类是取值128以上的是信息报文。

□代码：占1字节，标识对应ICMP报文的代码。它与类型字段一起共同标识了ICMP报文的详细类型。

□校验和：这是对包括ICMP报文数据部分在内的整个ICMP数据报的校验和，以检验报文在传输过程中是否出现了差错。其计算方法与我们在7.3.4节介绍的IP报头中的校验和计算方法是一样的。

□标识：占2字节，用于标识本ICMP进程，但仅适用于回显请求和应答ICMP报文，对于目标不可达ICMP报文和超时ICMP报文等，该字段值全为0。

3.常见的ICMP报文

下面是几种常见的ICMP报文。

(1) 响应请求

我们日常进行的Ping操作中就包括响应请求（类型字段值为8）和应答（类型字段值为0）ICMP报文。一台主机向一个节点发送一个类型字段值为8的ICMP报文，如果途中没有异常（如没有被路由器丢弃、目标不回应ICMP或传输失败），则目标返回类型字段值为0的ICMP报文，说明这台主机存在。

(2) 目标不可到达、源抑制和超时报文

这三种报文的格式是一样的。目标不可到达报文（类型字段值为3）在路由器或主机不能传递数据报时使用。例如我们要连接对方一个不存在的系统端口（端口号小于1024）时，将返回类型字段值为3、代码字段值为3的ICMP报文。常见的不可到达类型还有网络不可到达（代码字段值为0）、主机不可到达（代码字段值为1）、协议不可到达（代码字段值为2）等。

源抑制报文（类型字段值为4，代码字段值为0）则充当一个控制流量的角色，通知主机减少数据报流量。由于ICMP没有恢复传输的报文，所以只要停止该报文，主机就会逐渐恢复传输速率。最后，无连

接方式网络的问题就是数据报会丢失，或者长时间在网络游荡而找不到目标，或者拥塞导致主机在规定时间内无法重组数据报分段，这时就要触发ICMP超时报文的产生。

超时报文（类型字段值为11）的代码域有两种取值：代码字段值为0表示传输超时，代码字段值为1表示重组分段超时。

（3）时间戳请求

时间戳请求报文（类型值字段13）和时间戳应答报文（类型值字段14）用于测试两台主机之间数据报来回一次的传输时间。传输时，主机填充原始时间戳，接收方收到请求后填充接收时间戳后以类型值字段14的报文格式返回，发送方计算这个时间差。有些系统不响应这种报文。

7.3.12 IPv6协议簇中的其他协议

除了前面介绍的IPv6协议外，在IPv6协议簇中还包括ICMPv6、ND（邻居发现）和MLD（绝色播侦听器发现）协议。这些协议分别用来取代IPv4协议簇中的对应协议：

□ICMPv6取代ICMP。ICMPv6提供诊断功能，并可在IPv6数据报无法传送时报告错误。

□MLD取代IGMP。MLD管理IPv6多播组成员身份。

□ND取代ARPND管理相邻节点间的交互，包括自动配置地址和将下一跃点IPv6地址解析为MAC地址。

因为IPv6毕竟目前还没有正式全面使用，所以下面仅对这些协议进行简单介绍。

1.ICMPv6

ICMPv6与前面介绍的IPv4协议簇中的ICMP一样，用于报告传送或转发中的错误并为疑难解答提供简单的回显服务。同时ICMPv6协议还为ND和MLD消息提供消息结构。

表7-5所示为RFC 2463中定义的ICMPv6消息，并作了说明。

表 7-5 常见的 ICMPv6 消息

ICMPv6 消息	说 明
回显请求	发送主机发送请求回显消息来检查与特定节点的 IPv6 连接
回显应答	节点发送回显答复消息响应 ICMPv6 请求回显消息
无法到达目标	路由器或目标主机发送“无法到达目标”消息，通知发送主机它们的数据报或有效负载无法传送
数据报太大	路由器发送“数据报太大”消息，通知发送主机数据报因太大而无法转发
超时	路由器发送“超时”消息，通知发送主机 IPv6 数据报的跃点限制已到期
参数问题	路由器在处理 IPv6 标头或 IPv6 扩展标头时，如果遇到错误，便发送“参数问题”消息来通知发送主机

ICMPv6与IPv4协议簇中的ICMP一样，也包含了一系列已定义的“无法到达目标”消息，如表7-6所示。

表 7-6 常见的 ICMPv6 无法到达目标消息

无法到达目标消息	说 明
找不到路由	路由器在其本地 IPv6 路由表中找不到指向目标 IPv6 地址的路由时，便发送此消息
管理策略禁止通信	当路由器上配置的某条策略禁止与目标进行通信时，路由器便发送此消息。例如，当防火墙丢弃数据报时，路由器便发送此类消息
无法到达目标地址	IPv6 路由器无法解析目标的 MAC 地址时便发送此消息
目标端口无法访问	当发往目标 UDP 端口的、包含 UDP 消息的 IPv6 数据报与正在侦听的应用程序不对应时，目标主机便发送此消息

2.ND

ND是一组ICMPv6消息和过程，用于确定相邻结点间的关系。ND取代了IPv4中使用的ARP、ICMP路由器发现和ICMP重定向功能，提供了更丰富的功能。主机可以使用ND完成以下任务：

□发现相邻的路由器。

□发现并自动配置地址和其他配置参数。

路由器可以使用ND完成以下任务：

□公布它们的存在、主机地址和其他配置参数。

□向主机提示更好的下一跃点地址以帮助数据报转发到特定目标。

结点（包括主机和路由器）可以使用ND完成以下任务：

□解析IPv6数据报将被转发到的一个相邻结点的链路层地址（又称MAC地址）。

□动态公布MAC地址的更改。

□确定某个相邻结点是否仍然可以到达。

表7-7所示为RFC 2461解释文档中描述的ND过程。

表 7-7 IPv6 邻居发现过程

邻居发现过程	说 明
路由器发现	主机通过该过程来发现它的相邻路由器
前缀发现	主机通过该过程来发现本地子网目标的网络前缀
地址自动配置	无论是否存在地址配置服务器（例如运行动态主机配置协议版本 6（DHCPv6）的服务器），该过程都可以为接口配置 IPv6 地址
地址解析	结点通过该过程将邻居的 IPv6 地址解析为它的 MAC 地址。IPv6 中的地址解析相当于 IPv4 中的 ARP
下一跃点确定	结点根据目标地址通过该过程来确定数据报要转发到的下一跃点 IPv6 地址。下一跃点地址可能是目标地址，也可能是某个相邻路由器的地址
邻居不可访问性检测	结点通过该过程确定邻居的 IPv6 层是否能够发送或接收数据报
重复地址检测	结点通过该过程确定它打算使用的某个地址是否已被相邻节点占用
重定向功能	该过程提示主机更好的第一跃点 IPv6 地址来帮助数据报向目标传送

IPv6地址解析包括交换邻居请求和邻居公布消息，从而将下一跃点IPv6地址解析为其对应的MAC地址。发送主机在适当的接口上发送

一条多播邻居请求消息。邻居请求消息包括发送节点的**MAC**地址。

当目标节点接收到邻居请求消息后，将使用邻居请求消息中包含的源地址和**MAC**地址的条目更新其邻居缓存（相当于**ARP**缓存）。接着，目标节点向邻居请求消息的发送方发送一条包含它的**MAC**地址的单播邻居公布消息。

接收到来自目标的邻居公布后，发送主机根据其中包含的**MAC**地址使用目标节点条目来更新它的邻居缓存。此时，发送主机和邻居请求的目标就可以发送单播**IPv6**通信量了。

主机通过路由器发现过程会尝试发现本地子网上的路由器集合。**IPv6**路由器发现过程如下：

- 1) **IPv6**路由器定期在子网上发送多播路由器公布消息，以公布它们的路由器身份信息和其他配置参数（例如地址前缀和默认跃点限制）。

- 2) 本地子网上的**IPv6**主机接收路由器公布消息，并使用其内容来配置地址、默认路由器和其他配置参数。

- 3) 一个正在启动的主机发送多播路由器请求消息。收到路由器请求消息后，本地子网上的所有路由器都向发送路由器请求的主机发送

一条单播路由器公布消息。该主机接收路由器公布消息并使用其内容来配置地址、默认路由器和其他配置参数。

3.MLD

MLD是IPv4协议簇中的IGMP更新版本，是由路由器和节点交换的一组ICMPv6消息，供路由器用来为各个连接的接口发现有侦听节点的IPv6多播地址的集合。与IGMP一样，MLD只能发现那些至少包含一个侦听器的多播地址，而不能发现各个多播地址的单个多播侦听器的列表。与IGMP不同的是，MLD使用ICMPv6消息而不是定义它自己的消息结构。MLD消息有三种类型：

□多播侦听器查询：路由器使用多播侦听器查询消息来查询子网上是否有多播侦听器。

□多播侦听器报告：多播侦听器使用多播侦听器报告消息来报告它们有兴趣接收发往特定多播地址的多播通信量，或者使用这类消息来响应多播侦听器查询消息。

□多播侦听器完成：多播侦听器使用多播侦听器完成消息来报告它们可能是子网上最后的多播组成员。

7.4 路由和路由算法

网络层的主要功能就是将分组从源节点路由到目的节点中，而且在大多数计算机网络中，采用的是数据报分组交换方式，数据报分组需要经过多跳（Hop，也就是要经过多少个路由器）才能到达目的地的。

路由功能其实是一种数据报分组交换路径选择行为，是网络层的一种基本功能。路由功能与我们旅行或者运输货物选择最佳（注意，“最佳”并不代表最快，是要综合考虑的）路线是一个道理。图7-22所示的是我们假设要从北京发一封邮件到美国的旧金山朋友那里的示例。

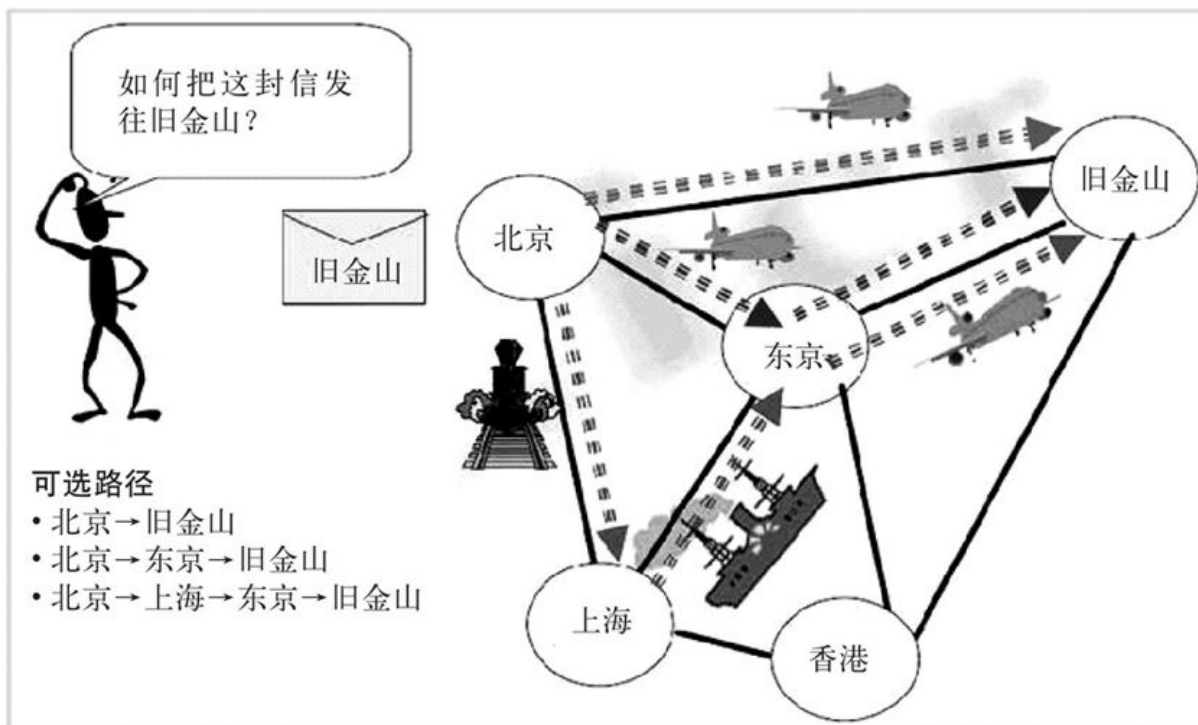


图 7-22 为“路由”功能打的一个比方

从图中可以看出，发这样一封信可以有多条路线，不同路线又需采用不同的邮寄方式、不同的线路长度和邮寄时间，当然运输成本也各不一样，最终选择哪条线路要综合总体邮寄成本、邮寄时间和途经邮局的可靠性等方面进行考虑。路由选择也是一样的，它要综合考虑许多因素的，如线路长度、信道带宽、线路的稳定性、途经端口的开销等。不同的路由算法所考虑的因素并不一样，这些将在本章后面介绍。

7.4.1 路由的分类

路由（**Routing**）是把信息从源节点通过网络传递到目的节点的行为，简单地讲路由就是指三层设备从一个接口上收到数据包，根据数据包的目的地址进行定向，并转发到另一个接口的过程。但在这条路由路径上，至少需要遇到一个中间结点，那就是提供路由功能的设备，如路由器和三层交换机。路由与桥接对比的主要区别在于，桥接发生在OSI参考协议的第二层（链接层），连接的是同一网络或同一子网的不同网段，而路由发生在第三层（网络层），连接的是不同网络或不同子网。

路由功能的实现是依靠路由器或三层交换机中的路由表进行的。路由又分静态路由（**Static Routing**）和动态路由（**Dynamic Routing**）两大类。下面分别予以介绍。

1.静态路由

静态路由是我们经常需要配置的，特别是在小型局域网中，因为它配置和管理都比较简单。总体来说，静态路由具有以下几个方面的特点。

（1）手动配置

静态路由需要管理员根据实际需要一条条自己手动配置，路由器不会自动生成所需的静态路由。静态路由中包括目的节点或目的网络

的IP地址，及数据包从当前路由器开始路由的第一个下一跳（通常就是网关）所对应的接口或IP地址。

（2）路由路径固定不变

因为静态路由是手动配置的，且静态的，所以当网络的拓扑结构或链路的状态发生变化时，这些静态路由不能自动修改，需要网络管理员手工去修改路由表中相关信息。

（3）不可通告性

静态路由信息在默认情况下是私有的，不会通告给其他路由器，也就是当在一个路由器上配置了某条静态路由时，它不会被通告网络中相连的其他路由器。但网络管理员可以通过重新发布静态路由为其他动态路由，使得网络中其他路由器也可获此静态路由。

（4）单向性

静态路由是具有单向性的，也就是它仅为数据提供沿着下一跳的方向进行路由，不提供反向路由。所以如果你想要使源节点与目的节点或网络进行双向通信，就必须同时配置回程静态路由。在现实应用中经常会遇到这样的问题，就是明明配置了到达某节点的静态路由，可还是Ping不通，其中一个重要原因就是没有配置回程静态路由。

如图7-23所示，如果想要使得PC1（PC1已配置了A节点的IP地址10.16.1.2/24作为网关地址）能够Ping通PC2，则必须同时配置以下两条静态路由，具体配置方法在此不作介绍。

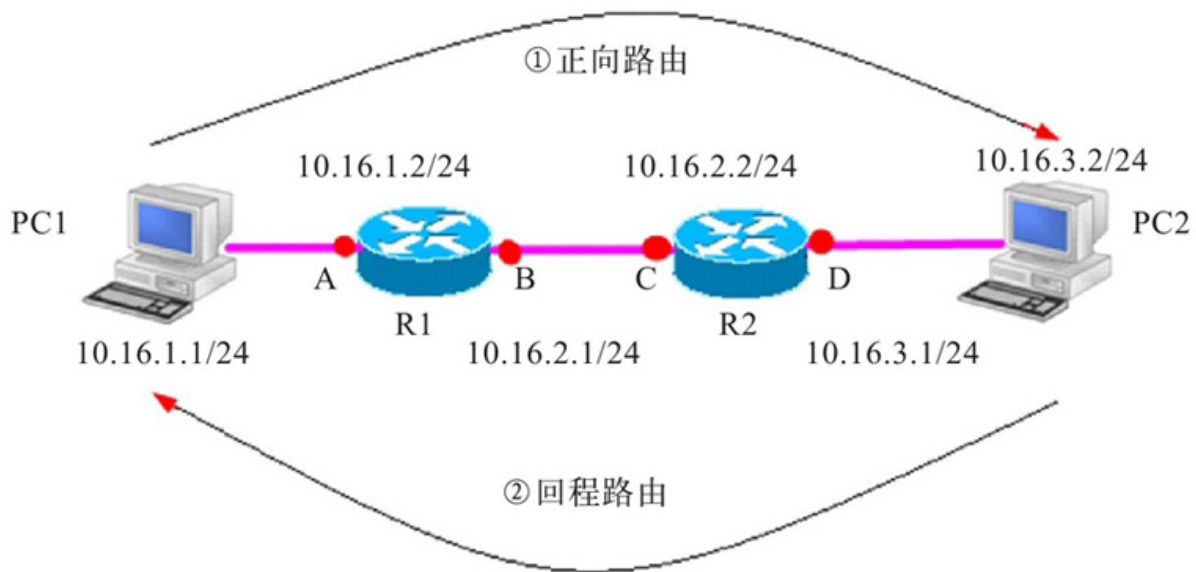


图 7-23 静态路由单向性示例

正向路由：在R1路由器上配置了到达PC2的正向静态路由（以PC2 10.16.3.2/24作为目的节点，以C结点IP地址10.16.2.2/24作为下一跳地址）；

回程路由：在R2路由器上配置一条到达PC1的回程静态路由（以PC1 10.16.1.1/24作为目的节点，以B结点IP地址10.16.2.1/24作为下一跳地址），以提供Ping过程回程ICMP消息的路由路径。

(5) 接力性

如果某条静态路由中间经过的跳数大于1（也就是整条路由路径经历了三个或以上路由器结点），则必须在除最后一个路由器外的其他路由器上依次配置到达相同目的节点或目的网络的静态路由，这就是静态路由的接力性，否则仅在源路由器上配置静态路由还是不可以的。

就像你要从长沙到北京去，假设中间要途经的站点包括：武汉-郑州-石家庄，可人家只告诉你目的地是北京，以及从长沙出发的下一站是武汉。对于一个没有多少旅游经验的人来说，你不可能知道到了武汉后又该如何走，必须有人告诉你到了武汉后再怎么走，到了郑州后又该怎么走，……这就是接力性。

图7-24所示是一个三个路由器串联的简单网络，各个路由器结点及PC机的IP地址均在图中进行了标注，PC1已配置好指向R1的A结点地址的网关，现假设要使PC1能Ping得通PC2，则需要配置以下四条路由（两条正向，两条回程）：

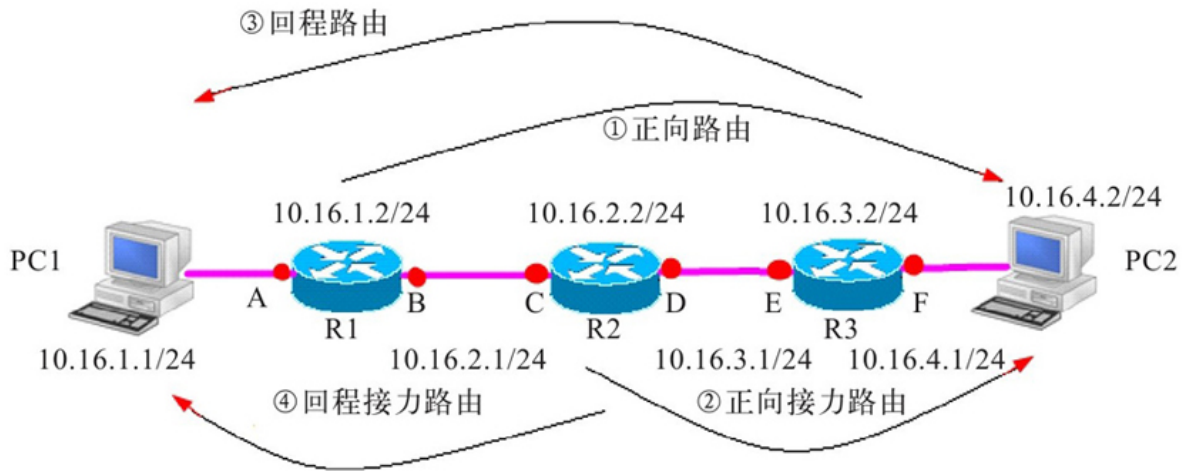


图 7-24 静态路由接力性示例

①：在R1路由器上配置了到达PC2的正向静态路由（以PC2 10.16.4.2/24作为目的结点，以C结点IP地址10.16.2.2/24作为下一跳地址）。

②：在R2路由器上配置了到达PC2的正向接力静态路由（同样以PC2 10.16.4.2/24作为目的节点，以E结点IP地址10.16.3.2/24作为下一跳地址）。

③：在R3路由器上配置一条到达PC1的回程静态路由（以PC1 10.16.1.1/24作为目的节点，以D结点IP地址10.16.3.1/24作为下一跳地址），以提供Ping过程回程ICMP消息的路由路径。

④：在R2路由器上配置一条到达PC1的回程接力静态路由（同样以PC1 10.16.1.1/24作为目的节点地址，以B结点IP地址10.16.2.1/24作为下一跳地址），以提供Ping过程回程ICMP消息的接力路由路径。

（6）优先级较高

因为静态路由明确指出了到达目的网络，或者目的节点的路由路径，所以在所有同目的地址的路由中，静态路由的优先级是除“直连路由”外最高的，也就是如果配置了到达某一网络或者某一结点的静态路由，则优先采用这条静态路由，只有当这条静态路由不可用时才会考虑选择其他的路由。

说明 在这里要特别注意一个方面，那就是默认路由的优先级。如果在一个路由器同时有一条目的地址相同的静态路由和一条默认路由，则首先选择的是静态路由。要使对应的默认路由起作用，就必须删除相同目的网络或目的节点的静态路由，否则会冲突，默认路由不起作用。

（7）适用小型网络

静态路由一般适用于比较简单的小型网络环境，因为在这样的环境中，网络管理员易于清楚地了解网络的拓扑结构，便于设置正确的路由信息。同时小型网络所需配置的静态路由条目不会太多。如果网络规模较大，拓扑结构比较复杂，则不宜采用静态路由，因为这样的配置工作量实在太大了。

2.动态路由

静态路由对于小型且变化不是很频繁的网络来说还是可行的，如局域网。但是对于较大型的广域网来说，由于拓扑结构较复杂，且网络结构可能经常变动，静态路由就不再适用了，通常采用更加灵活，更具自动特性的动态路由。总体来说，动态路由具有以下几个方面的特点：

（1）自动生成

动态路由的一个最重要特点就是在网络中某条路由所包括的路由器同时启动了某种动态路由协议，通告了各自所直接连接的网络后，则这些路由器间就会自动生成这些路由器直接连接的网络间的路由表项，管理员无须一一手动创建。这对于较大型的网络来说，是最方便、最简单的路由选择了。

（2）自动调整

当网络结构发生改变，我们手动的静态配置也往往无法及时跟着改变，但动态路由可以随时根据网络拓扑结构的变化调整路由表项，同时还会自动删除无效的动态路由表项，更加方便路由管理。

（3）自动通告

前面说了，一个路由器上的静态路由表项是一个路由器私有的，但是动态路由可以在相邻路由器上相互通告，以便及时反映拓扑结构

的变化，生成新的动态路由表项。

(4) 自动生成双向路由

虽然单条动态路由也是单向的，但是路由器在生成某条路径的动态路由时会自动生成回程路由表项，也就是会同时双向路由表项。

(5) 仅可生成网络间的路由表项

动态路由仅可生成各路由器所直接连接的各个网络或子网间的路由表项，不能生成到达具体节点或主机的动态路由表项，如果仅需要生成到达某个结点或主机的路由，则需要选择静态路由。

(6) 不同动态路由不兼容

动态路由根据所采用的路由算法的不同又可分为多种类型，如RIP（路由信息协议）、OSPF（开放最短路径优先）、EIGRP（增强内部网关路由协议）、IS-IS（中间系统到中间系统）、BGP（边界网关协议）等。不同的动态路由协议主要适用的网络环境不一样，也是不兼容的，但可以进行相互重发布，具体将在本章后面介绍。

7.4.2 路由算法基础

路由算法（**Routing Algorithm**）是在给定一组路由器及连接路由器链路的情况下，找出一条从源节点到目标节点的最佳路径。通俗地讲，就是把路由器选择最佳路径的策略称为路由算法，是路由器的关键功能所在。为了完成这项工作，在路由器中需要收集和保存着各种与传输路径的相关数据，如拓扑结构、端口度量、端口速率等，然后根据相应的路由算法生成一个个路由表（**RoutingTable**）表项，在数据包转发时提供路由选择。

从大的方面来说，路由算法可以分为非自适应路由算法（**Nonadaptive Algorithm**）和自适应路由算法（**Adaptive Algorithm**）两类。非自适应路由算法的典型代表就是我们常说的静态路由，但不仅如此。而动态路由中所采用的算法都属于自适应路由算法。

1.非自适应路由算法

非自适应路由算法是指那些不能根据网络流量和拓扑结构的变化更新路由表，仅使用静态路由表的路由算法，又称固定式路由选择算法。非自适应路由算法的特点是简单、开销少、灵活性差。它主要包括静态路由（**Static Routing**）、扩散（**Flooding**）法、随机走动

(Random Walk) 法、最短路径 (Shortest Path) 法、基于流量的路由 (Flow-based Routing) 法。下面分别予以简单介绍。

(1) 静态路由

静态路由一般不认为是真正的路由算法，它是由管理员在路由器上手动一条条创建的路由表项。在静态路由表项中包括目的节点或网络IP地址、下一跳IP地址或接口。具体参见上节介绍。我们通常所说的路由算法一般不包括静态路由算法，因为静态路由只是在开始路由前由管理人员手动建立的静态映射表项，不能对网络改变做出反映，通常被认为不适用于现在的大型、易变的网络。

(2) 扩散法

扩散法是当一个路由器接口收到一个报文分组后，即向它所有接口（包括接收该分组的源接口）进行复制扩散。因为有多条可能的路径，所以即使网络局部出现了故障也不影响通信，但大量重复分组加重了网络负担。正因如此，这种路由算法适宜于网络规模小、通信负载轻、可靠性要求极高的通信场合，如军用通信中。

(3) 随机走动法

随机走动法是当结点收到分组后，向所有与之相邻的结点中随机选择一个将分组转发出去。因为分组会在网络中乱窜，所以路由可

达的概率还比较高。这种方法虽然简单，但不是最佳路由，通信效率低，分组传输延迟也不可预测，实用价值低。

（4）最短路径法

一般来讲，即使网络节点直接相连，传输时延也各不一样，这与线路质量、网络结点忙与闲的状态，结点处理能力等很多因素有关。定量分析中，常用开销最小作为网络结点之间选择的依据，结点间的传输时延是决定费用的主要因素。

最短路径法是由Dijkstra提出的，其基本思想是：将源节点到网络中所有结点的最短通路都找出来，作为这个结点的路由表。当网络的拓扑结构不变、通信量平稳时，该点到网络内任何其他结点的最佳路径都在它的路由表中。如果每一个结点都生成和保存这样一张路由表，则整个网络通信都在最佳路径下进行。每个结点收到分组后，查表决定向哪个后继结点转发。

（5）基于流量的路由算法

最短路径算法是只考虑网络拓扑结构来寻找最短路径，没有考虑网络流量、负载对路由选择的影响，而基于流量的路由算法则结合了网络拓扑结构和通信流量两方面的因素进行路由选择。它需要先知道网络拓扑结构、结点之间的平均流量、各条线路的容量，然后在此基础上采用适当的选择算法，从而找出最佳路由。基于流量的路由算法

的基本原理是根据知道一条线路的负荷和平均流量，用排队计算出该线路的分组平均时延，再由所有线路的平均时延直接计算出流量加权平均值，从而得到整个网络的平均分组时延。此方法可使网络通信量更加平衡，得到较小的平均分组时延。

2.自适应路由算法

我们通常所说的动态路由均属于自适应路由算法类型，可根据网络流量和拓扑结构的变化更新路由表。特点是开销大、健壮性好和灵活性好。在这类路由算法中，路由器又是如何收集网络拓扑结构和端口信息，然后确定最佳路由呢？这要依据不同的动态路由协议所采用的动态路由算法类型而定。

总体来说，这些动态路由协议中，主要有两种动态路由算法：总体式路由算法和分散式路由算法。采用分散式路由算法时，每个路由器只与直接相连的路由器交换路由信息，每个路由器只有相邻路由器的路由信息，而没有网络中的其他路由器的路由信息，其代表就是“距离矢量”（Distance Vector, DV）算法。而采用总体式路由算法时，每个路由器都拥有网络中某个区域或者整个网络中所有其他路由器的全部信息以及网络的流量状态，其代表包括链路状态（Link State, LS）算法和分级路由（Hierarchical Routing）算法。

（1）距离矢量路由算法

距离矢量简单地讲就是有方向的距离，也就是对应路由条目中源和目的站点间的距离。这个距离就是指跳数（hop），或者叫度量（Metric），从本结点起，每经过一个路由器（也可以是提供路由功能的三层交换机，下同），就加一跳。也就是对应路由条目中除源站点所在网络直接连接的路由器外，到达目的站点所经过的路由器数（路径中所有路由器数减1）。图7-25所示为一个四台路由器串联的网络示例，图中从PC1到达PC2的路由条目的跳数（也就是距离）为3（共经过四个路由器，减1后得到3）。但是PC3到达PC2的跳数就只有1了（共经过2个路由器，减1后得到1），因为它们两个之间除了源路由器外，只经过了一个路由器。在这里要特别注意的是，跳数是针对所经过的路由器数而言的，而不是所经过的网络数而言的，因为连接在同一路由器的各个网络之间是没有跳数的，也即跳数为0。

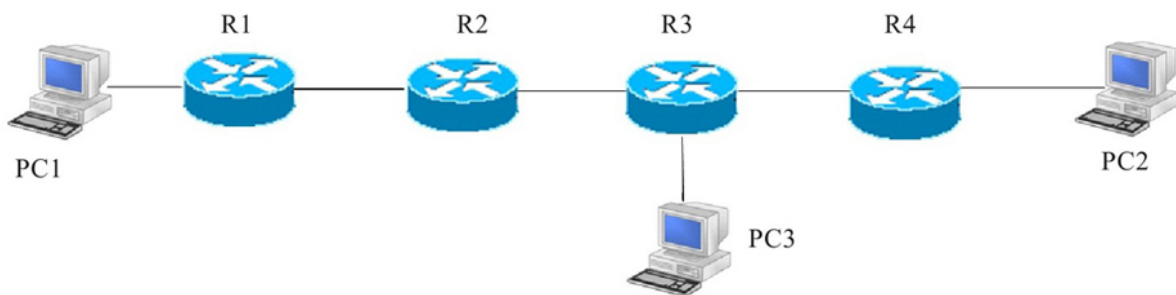


图 7-25 距离矢量计算示例

采用距离矢量算法的典型路由协议就是RIP，但它规定一条RIP路由中最多只能15跳，超过这个数就认为路由不可达。其他采用距离矢量算法的路由协议有IGRP（内部网关路由协议）、EIGRP（增强型内

部网关路由协议)和BGP(边界网关协议),但EIGRP同时还支持“链路状态”路由算法。IGRP和EIGRP的最大跳数均为255,比RIP的15大了许多,BGP更没有最大跳数限制,所以可以适用于更大的网络。

(2) 链路状态路由算法

链路状态算法(又称最短路径优先算法)比较复杂,它不仅是要根据所经过的路由器多少,还要根据路径中各段链路的状态来计算最佳路由路径。而这个链路状态包括:接口的IP地址、子网掩码、网络类型(如以太网链路或串行点对点链路)、该链路的端口开销(Cost)、该链路上的所有的相邻路由器。而端口开销又与端口的接入带宽有关。

采用链路状态路由算法的动态路由协议主要包括OSPF(开放最短路径优先)、IS-IS(中间系统到中间系统)、EIGRP(同时支持“链路状态”和“距离矢量”两种算法)。

总体而言,链路状态路由协议是层次式的,网络中的路由器并不向邻居路由器传递路由表项,只是向邻居路由器通告它的一些链路状态。在这类路由协议网络中,会把网络中的所有路由器分成区域,收集区域的所有路由器的链路状态信息,然后根据状态信息生成网络拓扑结构,最后每一个路由器再根据拓扑结构计算出路由。而距离矢量路由协议是平面式的,所有的路由学习完全依靠邻居路由器,交换的

是完整的路由表项。所以链路状态路由协议更灵活，更适合大型的网络。

以上介绍的距离矢量和链路状态路由算法其实只是针对一个特定的区域而言的，还有一种是针对整个网络来说的路由算法，那就是下面将要介绍的分级路由算法。

(3) 分级路由算法

在前面介绍的距离矢量和链路状态路由算法中，每个路由器都需要保存其他路由器的一些信息。随着网络规模的扩大，网络中的路由器也将增加。因此，路由表的规模也将增大，从而使路由器不能有效地处理网络流量。使用分级路由（**Hierarchical Routing**）算法可以解决这个问题。

在分级路由算法中，路由器被分成很多组，称为区域。每个路由器都只有自己所在区域路由器的信息，而没有其他区域路由器的信息。所以在其路由表中，路由器只需要存储其他每个区域的一条记录。应用这种路由算法的典型路由协议包括**OSPF**、**IS-IS**、**BGP**。而在前面已说到，这些路由协议已采用了距离矢量或链路状态路由算法，这只是针对区域内部而言，在整个网络中，还是要采用分级路由算法的。

以上这些路由算法工作原理将在本章再具体介绍。

7.4.3 路由表基础

路由表（**Routing Table**）是一个存储在路由器或者联网计算机中的电子表格（文件）或类似的数据库。路由表存储着指向特定网络地址的路径（在有些情况下，还记录有路径的路由度量值），含有网络周边的拓扑信息。路由表建立的主要目标是为了实现路由协议和静态路由选择。路由器通常依靠所建立及维护的路由表来决定如何转发。路由表能力是指路由表内所容纳路由表项数量的极限。

无论是静态路由，还是各种动态路由，都会在路由器的路由表中保存，但是不同类型的路由表所显示的信息不完全相同。由设备管理员事先手动创建的固定路由表称为静态路由表，它不会随未来网络结构的改变而改变。动态路由表是路由器根据所选择的动态路由协议算法自动生成的路由表。

要特别注意的是，除了前面所说的静态路由和动态路由外，还有一种路由——直连路由，在这种路由器上直接连接网络（不表示具体主机的路由）。因为连接在同一个路由器上的各个网络都是相通的，无须配置任何其它路由条目，只需要在路由器上进行通告即可。

在路由器的路由表中，不同路由的代码（**Codes**）是不一样的。在Cisco路由器上只需要执行`show ip route`命令即可显示当前路由器的路

由表中所有路由表项，并且在最前面显示了不同路由的代码，C代表直连路由，S代表静态路由，I代表IGRP路由，B代表BGP路由，O代表OSPF路由，R代表RIP路由等。另外“S*”代表的是默认路由（它的目的地址是为任意地址0.0.0.0/0），也是静态路由的一种。具体如下。

```
R1#show ip route
Codes:C-connected,S-static,I-IGRP,R-RIP,M-mobile,B-BGP
D-EIGRP,EX-EIGRP external,O-OSPF,IA-OSPF inter area
N1-OSPF NSSA external type 1,N2-OSPF NSSA external type 2
E1-OSPF external type 1,E2-OSPF external type 2,E-EGP
i-IS-IS,L1-IS-IS level-1,L2-IS-IS level-2,*-candidate default
U-per-user static route,o-ODR
Gateway of last resort is 0.0.0.0 to network 0.0.0.0
192.168.10.0/30 is subnetted,1 subnets
C 192.168.10.0 is directly connected,Serial3/0
192.168.20.0/30 is subnetted,1 subnets
C 192.168.20.0 is directly connected,Serial3/3
172.31.0.0/24 is subnetted,1 subnets
S 172.31.10.0 [1/0] via 192.168.20.2,Serial3/3
S* 0.0.0.0/0 is directly connected,Serial3/0
```

每个静态路由表项至少包括以下信息：

□网络ID：就是目的地址的网络IP地址。

□子网掩码（subnet mask）：用来判断目的网络IP所属的网络，一般是以地址前缀长度来表示的，如上面示例中的192.168.10.0/30，表示掩码长度为30位，对应的子网掩码就为255.255.255.252。

□下一跳地址/接口（Next hop/interface）：就是数据在发送到目标地址的“旅途”中下一站的地址。

其他类型的路由表项的显示各不相同，具体大家可以参见笔者编著的《路由器配置与管理完全手册——Cisco篇》和《路由器配置与管理完全手册——H3C篇》。

经验之谈 路由表是适用于整个路由器的，而不是仅适用于具体的个别路由器接口的。也就是说，路由表中的每一个表项都适用于来自本路由器上任何接口，符合路由策略特性的数据包进行路由，而不只是适用于对来自某个具体的接口上的数据包进行路由。所以我们在Cisco路由器中为路由器配置路由时，是在“全局配置模式”或“路由器配置模式”下进行的，在华为、H3C路由器中是在“系统视图”下进行，都不是在具体的“接口配置模式”（或视图）下进行的。这一点许多读者都没有意识到。

7.4.4 路由优先级

通过前面的学习，我们已经知道，不同路由协议（包括静态路由协议和各种动态路由协议）所采用的路由算法是不一样的，如RIP协议是根据路径传递的跳数来决定路径长短（也就是传输距离）的，而EIGRP协议是根据路径传输中的带宽和延迟来决定路径开销来体现传输距离的，这是两种不同单位的度量值。那当路由器同时配置了源和目的节点（或目的网络）的不同类型路由时，路由器该首选采用哪条路由表项呢？如果没有一个统一的度量单位，就无法比较了，于是Cisco就定义了一种称为“管理距离”（Administrative Distance，AD）的度量单位，也是表示路由优先级的度量单位。

管理距离是指一种路由协议的路由可信度，是一个0~255之间的一个整数，值越低，优先级越高，也就是可信度越高。每一种路由协议按可靠性从高到低，依次分配一个管理距离，以代表对应类型路由的信任等级。这样我们就可以统一单位，从而衡量不同协议的路径开销以选出最优路径。Cisco设备中对不同协议路由所设定的默认管理距离如表7-8所示。注意：如果静态路由使用下一跳所对应的接口替代下一跳IP地址时，则目的网络被认为是直连网络，即管理距离为0。

表 7-8 Cisco 设备不同协议路由的默认管理距离

路由类型	管理距离	路由类型	管理距离
直连路由	0	OSPF	110
静态路由	1	IS-IS	115
EIGRP 汇总路由	5	RIP	120
外部 BGP	20	EGP	140
EIGRP	90	外部 EIGRP	170
IGRP	100	内部 BGP	200

但要，不同设备厂商对路由优先级的定义并不完全一样，如华为、H3C对不同协议路由所设置的默认优先级如表7-9所示。

表 7-9 华为、H3C 设备不同协议路由的默认优先级

路由类型	管理距离	路由类型	管理距离
直连路由	0	OSPF NSSA(非纯末梢区域)	150
OSPF	10	内部 BGP	255
IS-IS	15	外部 BGP	255
静态路由	60	未知路由	256
OSPF ASE(自治系统外部)	150		

以上仅是默认的优先级设置，管理员可以针对每条路由修改为不同于默认值的优先级值。通过对到达同一目的节点（或目的网络）的多条相同类型的路由配置不同的优先级值（在静态路由中称之为“浮动静态路由”）就可以达到“热备切换”的目的。就是当优先级较高的当前路由失效时，优先级较低的同一路由马上可以接替原来路由的工作，以确保到达某一目的地的路由始终有效。

7.4.5 路由算法设计目标和设计考虑

路由算法通常具有下列设计目标的一个或多个，即正确性（Correctness）、简单性（Simplicity）、健壮性（Robustness）、稳定性（Stability）、公平性（Fairness）、最优性（Optimality）和灵活有效性（Efficiency）。

（1）正确性和简单性

对于正确性和简单性比较好理解，就是要求所设计的路由算法能正确计算出网络中的最优路径，而且要求是最简单的，太复杂的会严重消耗路由器资源。换句话说，路由协议必须高效地提供其路由功能，尽量减少软件和应用的开销。当实现路由算法的软件必须运行在物理资源有限的计算机上时高效尤其重要。

（2）健壮性和稳定性

健壮性和稳定性就是要求路由算法在出现某些设备工作不正常，或不可预见的软件事件的情况下必须仍能正常处理，不会产生大的网络振荡。因为网络一旦投入运行，特别是大型公共服务网络，会要求持续、有效运行数年，甚至更久。在此期间，将会有各种各样的软、硬件问题。主机、路由器和线路都有可能会经常性地出错，网络拓扑结构也可能会多次发生变化。路由算法应能够很好地适应这些变化，

实现快速地收敛，而且不会要求所有主机都停止工作，即使在某台路由器崩溃时。

（3）公平性和最优性

公平性是要求路由算法对网络中的所有用户必须是平等的。最优性指路由算法选择最佳路径的能力，根据**Metric**的值和权值来计算。例如有一种路由算法可能使用跳数和延迟，但延迟的权值可能要大些。当然，路由协议必须严格定义计算**Metric**的算法。表面上看来这两者是有些冲突的，但事实上一个好的路由算法必须在两个指标之间选择一个最佳的平衡点。

（4）快速收敛性

快速收敛性是指路由算法必须能快速聚合，聚合是所有路由器对最佳路径达成一致的过程。当某网络事件使路径断掉或不可用时，路由器通过网络分发路由更新信息，促使最佳路径的重新计算，最终使所有路由器达成一致。聚合很慢的路由算法可能会产生路由环或网路中断。

在图7-26所示的简单路由结构中，某分组X在时间t1内到达路由器1，路由器1已经更新，并知道到达目的的最佳路径是以路由器2为下一跳的，于是就把该分组转发给路由器2。但是如果由于没有快速链路聚合性能，路由器2则可能还没有及时更新，它仍采用它上次数据发送的

路由表，认为它的最佳下一跳是路由器1，于是把该分组发回给路由器1，结果分组在两个路由器间来回传递，直到路由器2收到路由更新信息，或分组超过了生存期。

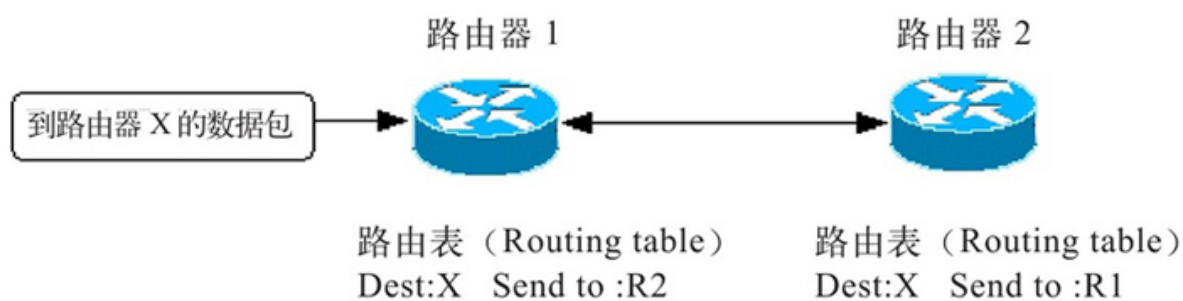


图 7-26 路由器不能快速聚合链路的示例

(5) 灵活有效性

路由算法还应该是灵活有效的，即它们应该迅速、准确地适应各种网络环境。例如，假定某网段断掉了，当知道问题后，很多路由算法对通常使用该网段的路径迅速选择次佳的路径。路由算法可以设计得适应网络带宽、路由器队列大小和网络延迟。

在设计路由算法时，还需要考虑许多路由技术元素，如性能标准、成本、决策时间、决策地点、网络信息源和更新时间等。性能标准方面需要考虑跳数、成本、延迟、吞吐量等几个方面。成本与数据率有关（数据率越高，成本越低），成本也与当前排队延迟有关。可采取最小跳计数标准，或者最小成本标准。

在决策时间方面，对于内部数据报，需要为每个包单独做路由决策；对于内部虚电路，只需在建立虚电路时做路由决策。在决策地点方面，对于分布式路由方式，每个结点都负责为到达的包选择一条输出链路，故需要考虑每个结点；对于集中式路由方式，则由某些指定结点（如网络控制中心）负责决策，所以只需针对中心结点；对于源路由方式，路由决策实际上由源站点而非网络作出，所以只需针对原始结点。

在网络信息源和更新时间这两个方面，大多数路由决策要求根据网络拓扑、通信负载和链路成本等知识做出决定。信息更新时间是信息源和路由决策的函数。如果像扩散算法那样是无信息的，也就无更新之说了；如果信息是局部的，则更新基本上是连续的；如果是相邻接结点或者全部结点，则在采用非自适应算法时从不更新信息，而在采用自适应算法时经常更新信息。

7.5 几种主要的路由算法解析

通过前面的学习我们知道，路由算法分为静态路由算法和动态路由算法两大类。其中静态路由算法包括最短路径路由（Shortest Path Routing）算法、扩散路由（Flooding Routing）算法和基于流量的路由（Flow-Based Routing）算法。动态路由算法则包括距离矢量路由（Distance Vector Routing）算法、链路状态路由（Link State Routing）算法和分级路由（Hierarchical Routing）算法等。本节仅对最短路径路由、扩散路由、距离向量路由和链路状态路由这四种算法进行介绍。

7.5.1 最短路径路由算法

最短路径路由算法是一种从源节点一段段地向目的节点方向查找到达源节点长度最短的链路，以此来确定一个结点到另一结点的最短路径。它的基本思想是构建通信子网拓扑图（不包括资源子网部分），图中的每个结点代表一个路由器，这些路由器串接起来就代表一条通信线路。

最典型的最短路径路由算法就是Dijkstra算法（迪克斯特拉算法），是由Dijkstra发明的。在该算法中，为了在一对给定的路由器结点之间选择一条最短路由路径，只需在通信子网拓扑图中找到在起始

和结束结点之间的中间结点串连起来后链路长度最短的路径（这里指线路长度）即可。它把最短路由的结点标识为工作结点，并且是永久性的结点，其到达源节点的距离值是不能改变的，其他的标识为临时性的结点，其到达源节点的距离可能会随工作结点的不同而改变。所有工作结点串联起来就是源节点和目的节点之间的最短路由路径，打个比方来说。现假设我们要从A城市到达H城市，如图7-27所示。首先我们肯定是先在A城市找一条与目的城市H同方向的最短出发路线（从图中可以看出共有三条路可选），假设由A到B这条路是最短的，所以首先选择这条路线出发。

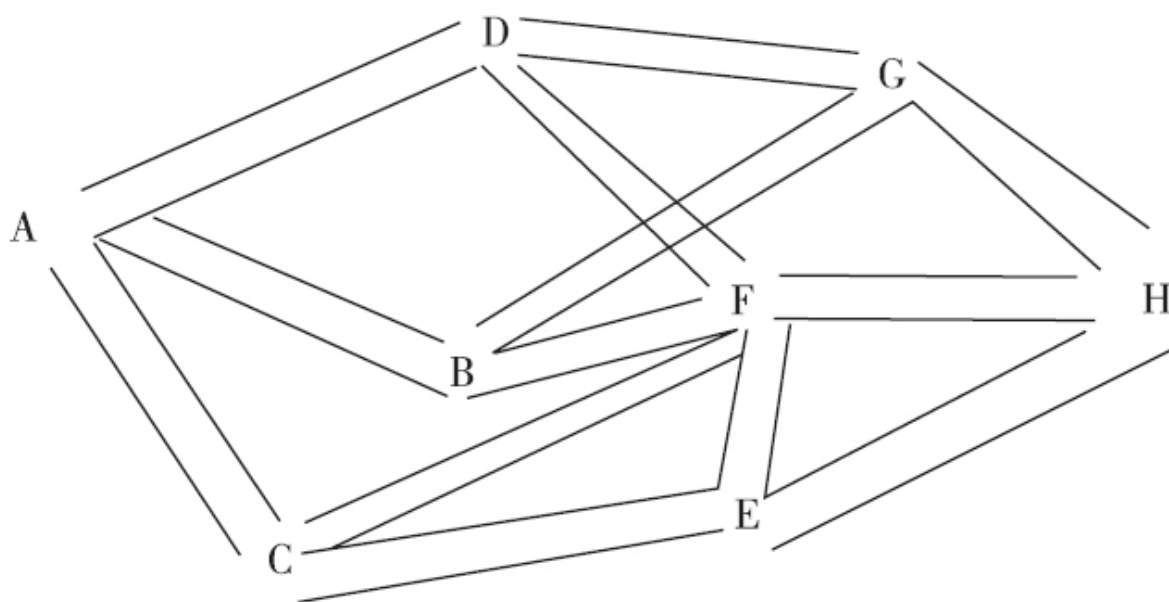


图 7-27 “Dijkstra算法”比喻示例

当我们到达B地点后，又要选择一条可以到达目的城市H的最佳路径（共有2条路线可选），这条路径要求离B地点最近，假设选择到达F

地点这条。然后在F地继续寻找一条可以到达目的城市H的最短路线，因为F地点有条路直接到达目的城市H，所以直接选择这条即可，最终到达了目的城市H。

以上就是Dijkstra最短路径算法的基本思想，下面通过一个具体的网络路由示例来加深对它的理解。

图7-28所示的子网图是一个典型的最短路径路由算法子网图，图中的每一个结点代表一台路由器，每条线段代表一条通信链路。现假设要使用Dijkstra算法计算结点A到结点D之间的最短路径。在网络中路由器启动时，首先需要初始化，测量每条链路的长度，参见图7-28所示的各条线段上的数字。

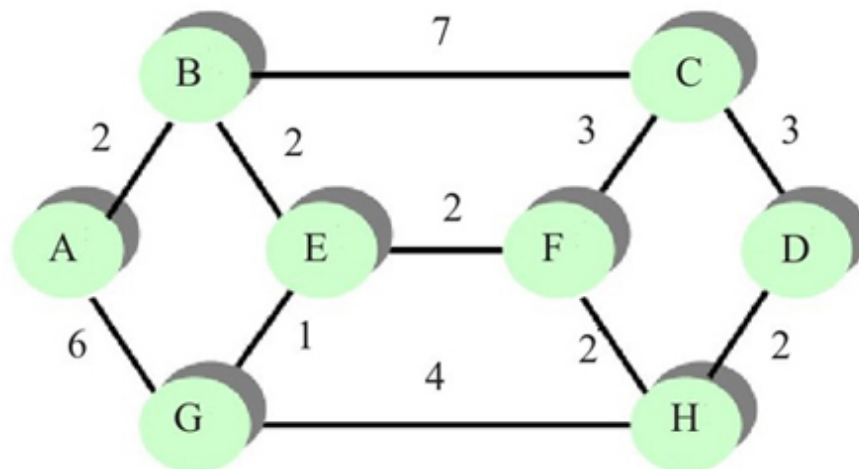


图 7-28 典型的Dijkstra算法子网图

下面是从A结点到到达D结点的路由确定步骤。

1) 首先将源结点A标记为永久性工作结点（我们用箭头来特别标识），然后依次检查每一个与A结点直接连接的相邻结点，并且把它们与A结点之间的距离重新以 (n, N) 的方式进行标识，其中的 n 为与A结点相距的链路长度， N 为最近的工作结点。

因为本示例中与结点A直接相邻的结点只有B和G，所以仅需标识这两个结点与A结点之间的距离。此时的工作结点为A，如图7-29所示，B结点的标识为 $(2, A)$ ，G结点的标识为 $(6, A)$ 。之所以这样标注，是因为B结点到A结点的链路长度为2，G结点到A结点的链路长度为6。其他与A结点不相邻的结点的距离标识为无穷远。

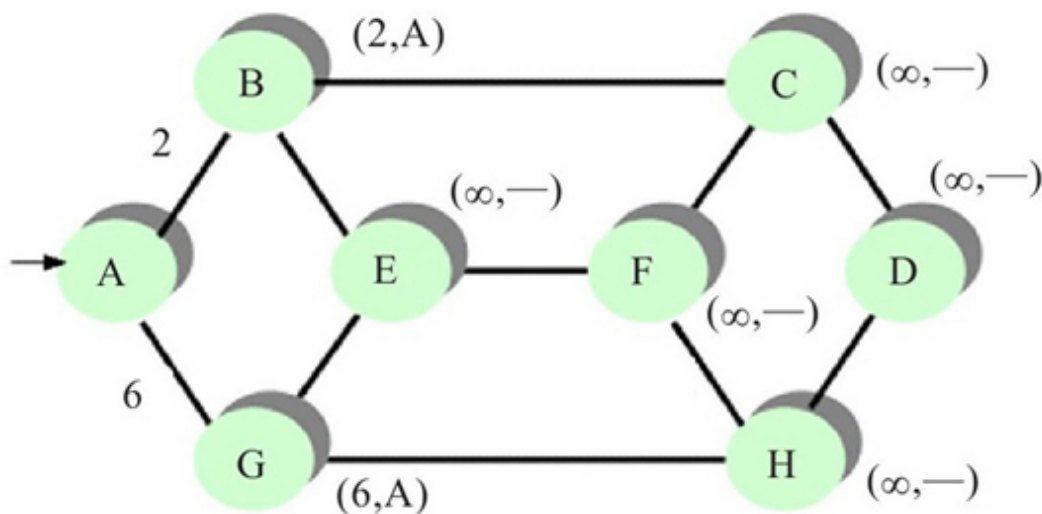


图 7-29 以A结点为工作结点标记其他相邻结点与A结点之间的距离

2) 比较B和G这两个结点与A结点之间距离，可以看出B结点的距离更短，于是把B结点改为工作结点（箭头移到B），同时将其变为永久性结点，其他结点（包括G结点）标注为临时结点。然后再以B结点

为工作结点，标记直接相邻的结点到源结点A的距离，当然对于前面已经计算过的结点将略过，如G结点。

本示例中与B结点直接相邻的结点中，除了A结点外还有C、E这两个结点。C结点到达A结点的距离就是C结点到B结点的链路长度7再加上B结点到A结点的链路长度2，所以C结点到A结点的距离为 $2+7=9$ ，故将其标识为 $(9, B)$ 。同理，E结点到A结点的距离为 $2+2=4$ ，故将其标识为 $(4, B)$ ，如图7-30所示。其他既不与A结点又不与B结点相邻的结点距离仍为无穷远。

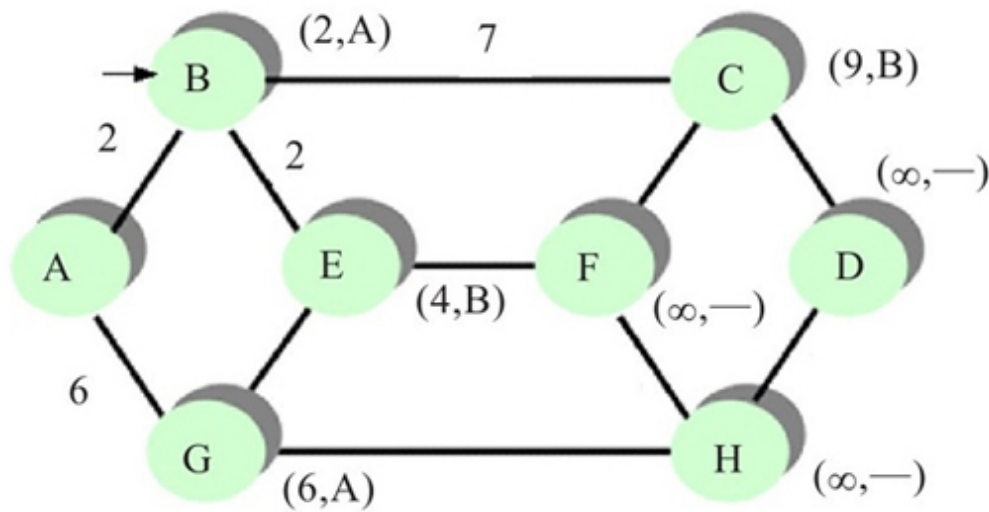


图 7-30 以B结点为工作结点标记其他结点与A结点之间的距离

3) 同样经过比较得出，E结点到A结点之间的距离（为4）比C结点到A结点的距离（为9）近，所以此时把E结点改为工作结点（箭头移到E），同时标注E结点为永久性结点，其他结点（包括C结点）标注为临时结点。

按同样方法标注与E结点直接相邻的结点（包括结点B、结点G和结点F）到E结点的距离，但对于前面已计算过的永久性B结点不再重新计算，而对于虽然原来已计算过，但为临时性结点的G以及F结点均需要重新计算。最终G结点的标识改为（5，E）（在此步以前为（6，A）），F结点标识为（6，E），表示G结点和F结点到达A结点的距离分别为5和6，如图7-31所示。

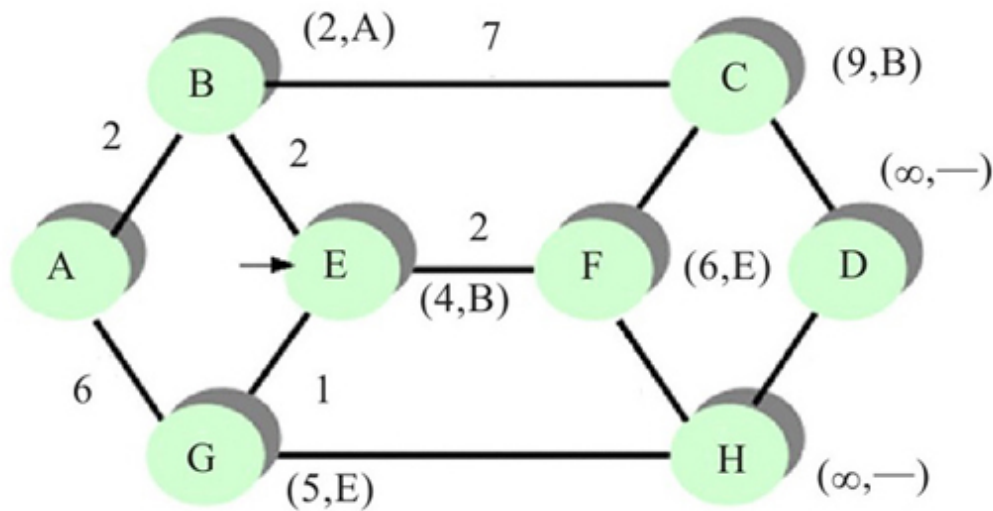


图 7-31 以E结点为工作结点标记其他结点与A结点之间的距离

4) 再用同样的方法比较G结点和F结点到达A结点之间的距离，可以得出G结点更近，所以此时把G结点改为工作结点（箭头移到E），同时标注G结点为永久性结点，其他结点（包括F结点）标注为临时结点，如图7-32所示。

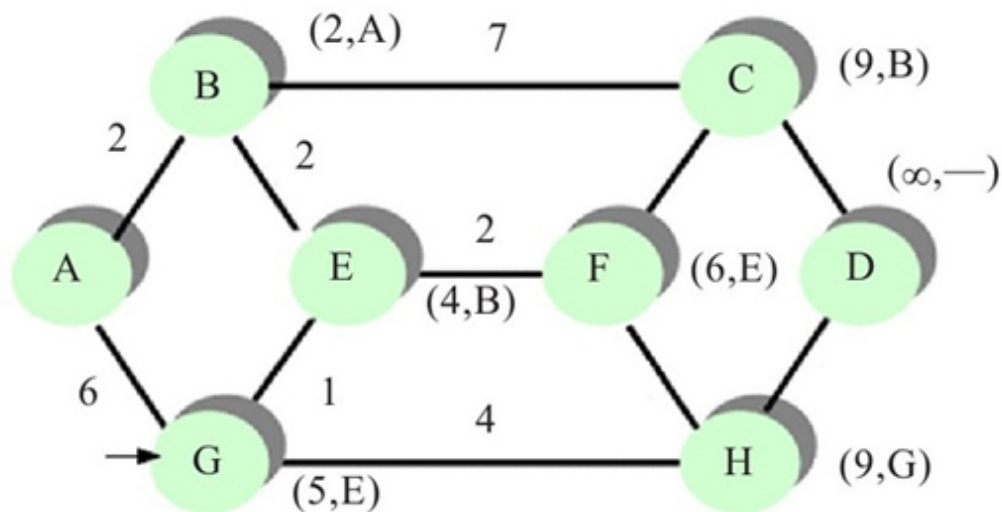


图 7-32 以G结点为工作结点标记其他结点与A结点之间的距离

再看一下与G结点直接相邻的结点，包括A、E、H这三个结点，但是A、E这两个结点在前面都已标识为永久性结点了，标识是不能更改的，所以在这里只需对H结点计算到达A结点的距离了。经过计算得出为 $(9, G)$ 。

在这里就要出现问题了，按照上面的计算，此时应该把H结点标识为下一个工作结点，但事实上，由H结点经G结点到达E结点的距离 $(4+1=5)$ 要长于由H结点经F结点到达E结点的距离 $(2+2=4)$ ，所以经过后面的计算发现，在前面把G结点标识为永久结点是错误的，这时要把F结点标识为工作结点（箭头移到F），撤销G结点永久工作结点的资格，如图7-33所示。

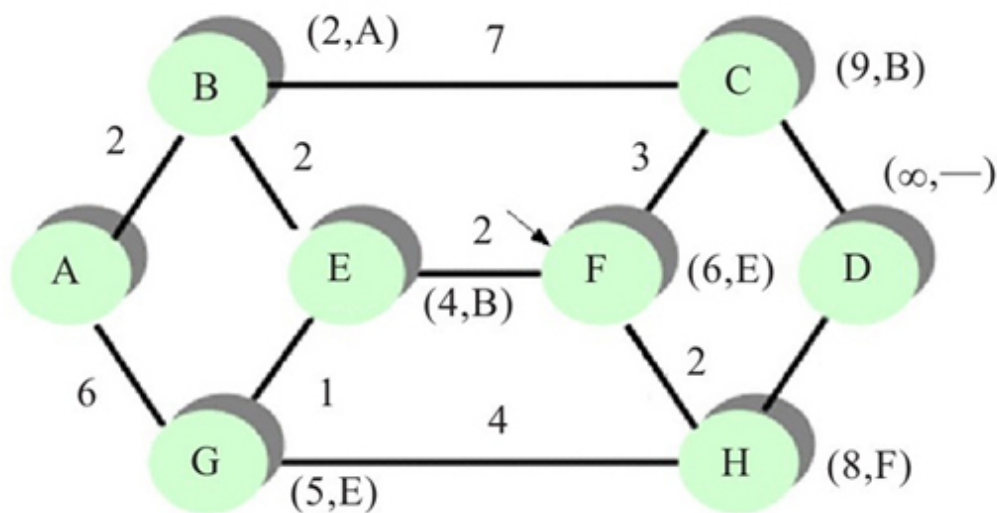


图 7-33 以F结点为工作结点标记其他结点与A结点之间的距离

5) 再检查与F结点相接相连接的相邻结点，除了原来已标识为永久性结点的E外，其余就是C和H这两个临时结点了。重新计算它们到结点A间的距离，得到的值分别为9和8。这里还要注意一个现象，就是对于C结点，本来属于临时结点，需要重新计算距离值，可是经过F结点到A结点的距离与原来计算所得的经过B结点到达A结点的距离是一样的，所以距离值不需要改变。此时把距离较短的H结点标识为工作结点（箭头移到H）和永久性结点，如图7-34所示。

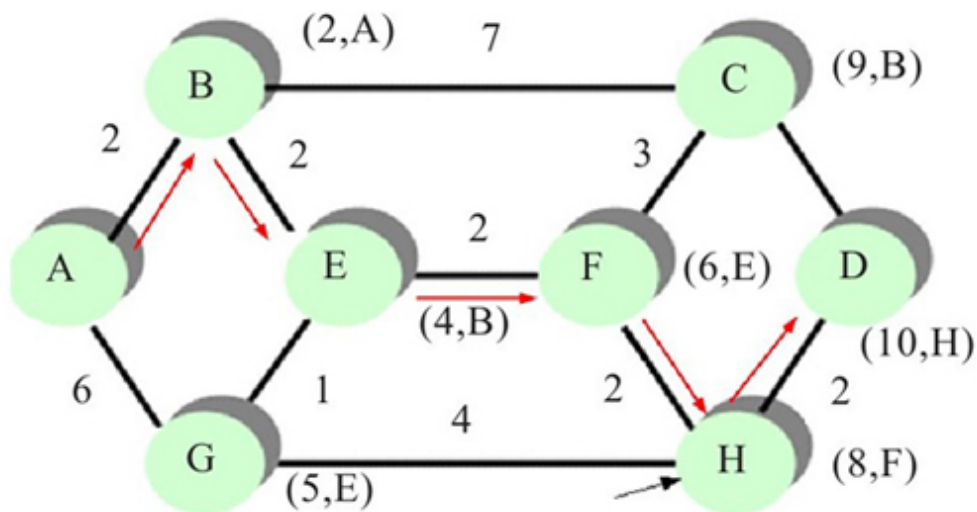


图 7-34 以H结点为工作结点标记其他结点与A结点之间的距离

此时，因为H结点是直接与目的结点D相连，所以无需再进行选举了，直接标识D结点的距离为（10，H）。即从源结点A到目标结点D的最短距离就为10，即 $2+2+2+2+2$ ，如图7-34所示的连线： $A \rightarrow B \rightarrow E \rightarrow F \rightarrow H \rightarrow D$ 。这样，找出了源结点到目的结点的最短距离了。

从以上可以看出，Dijkstra算法虽然能得出最短路径，但由于它遍历计算的结点很多，所以效率低。另外，有些结点还不能一次标识正确，因为还要考虑后续结点到达源结点的距离，如以上示例中G结点和F结点的工作结点标识，最初的标识就是错误的，因为它没有考虑后续结点到源结点的距离。

7.5.2 扩散算法

扩散（Flooding）算法也是一种静态路由算法。最初原始的扩散方法是，每一个接收到的分组都被从除接收端口之外的所有其他端口上转发出去。图7-35所示的是数据分组在源结点1上分散到了三个链路。

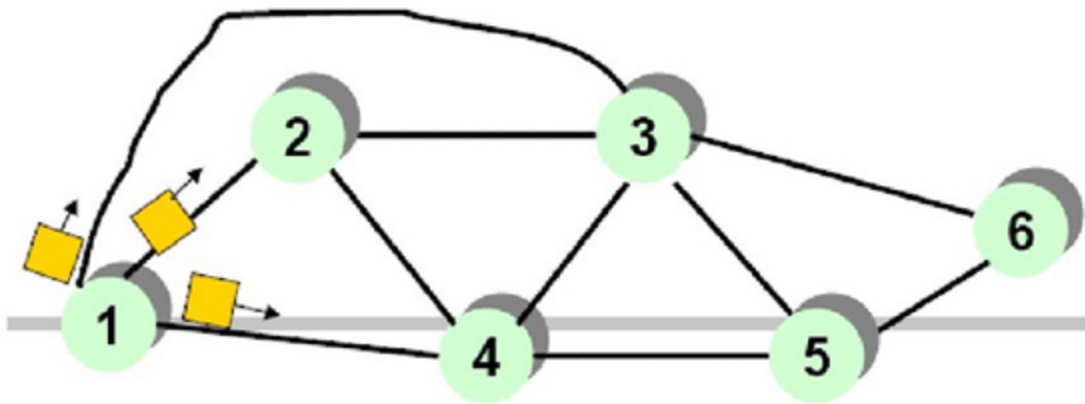


图 7-35 结点1分散成三个线路发送数据包

很显然，原始的扩散算法会产生大量的分组副本，如在图7-36中，当数据分组到了结点2和结点3后都会向所有可能的链路上发送分组，这样就会出现分组重复。而且越后面的结点收到的重复数据分组越多。如3号结点，它既有直接从结点1发来的分组，又有经过2号结点转发的同一分组，还有从4号结点发来的同一分组。不仅如此，这种扩散法还会造成分组死循环，如3号结点发来的分组，它又会向2号结点

返回一样的分组。这样，即使是真正的目的节点，也可能会接收到多个同样的分组，这显然是没有任何意义。

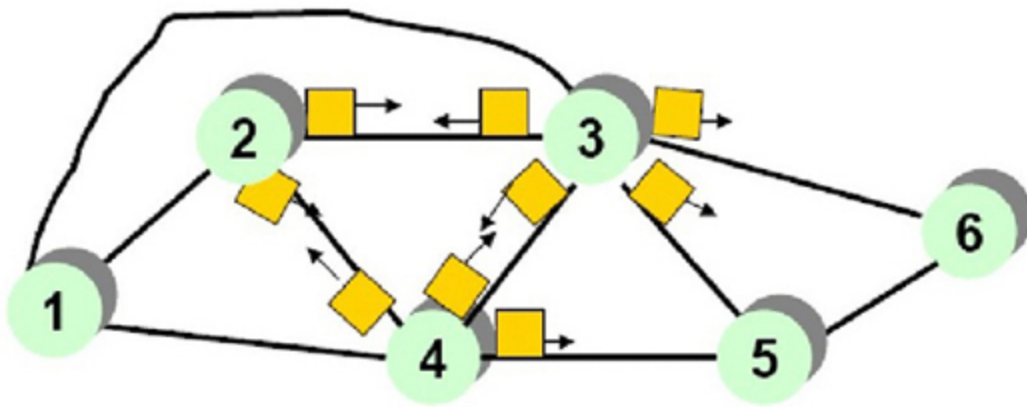


图 7-36 结点2和结点3向所有出镜线路发送数据包

为了解决以上各端口会重复接收相同分组的问题，设计者又想出了两种办法。其中一种办法是在每个分组头部中携带一个跳数计数器，分组每到一个结点其跳数计数器就减1，当计数器减到0时则该分组将被丢弃。而计数器的初始值可以设为通信子网的直径，即相距最远的两个结点之间的跳数，这样可以减少那些循环转发的分组。

另一种办法是在结点上记住哪些分组已经转发过了，对于重复接收的分组直接丢弃，从而确保一个分组不会被同一个结点转发两次。这要求源结点从主机收到一个分组后，为分组分配一个序号，同一分组的序号相同，就像我们在前面介绍数据帧的发送时为每个帧分配一个序号一样，重发的同一数据帧序号一样，这样接收端就可以确保对于同一个帧只接收一次。同时每一个结点对于每一个源结点都要维护

一张序号表，记录收到的来自同一源结点的所有分组序号。每当一个结点收到来自某个源结点的分组时，就用分组的序号去查找该源结点的序号表，如果该序号已在表中则该分组被丢弃。为了防止序号表过大，序号表中还应增设一个计数器 k ，表示序号直至 k 的分组都已经转发过了，从而不需要保留序号小于 k 的序号。

尽管扩散路由算法有它的不足，但也不是说它一无用处。由于它的数据分组的分散传送，使它的健壮性非常好，在一些经常遭到破坏的网络系统中是非常需要的。如军事网络系统，可能在战争中被瞬间炸得粉碎，但由于路由器间采取了扩散路由算法，这样在系统中只要存在一个链路，数据仍能完整地向前传输。另外，在无线网络中，一个站点发出的所有消息可以被位于无限距离范围内的所有站接收，这实际上也是一种扩散算法。还有就是在数据库系统中，如果数据库系统中有多台服务器，采取扩散算法，则数据库系统的更新效率会大大提高。

7.5.3 距离矢量路由算法

现代计算机网络通常使用动态路由算法，因为这类算法能够适应网络的拓扑和流量变化，其中最流行的两种动态路由算法是距离矢量路由算法和链路状态路由算法。

距离矢量路由算法（Distance Vector Routing, DV）是ARPANET网络上最早使用的路由算法，又称Bellman-Ford路由算法和Ford-Fulkerson算法，主要在RIP（Route Information Protocol）协议中使用。Cisco的IGRP和EIGRP路由协议也是采用DV这种路由算法的。

距离矢量路由算法的基本思想如下：每个路由器维护一个距离矢量（通常是以延时是作变量的）表，然后通过相邻路由器之间的距离矢量通告进行距离矢量表的更新。每个距离矢量表项包括两部分：到达目的节点的最佳输出线路，和到达目的节点所需时间或距离，通信子网中的其他每个路由器在表中占据一个表项，并作为该表项的索引。每隔一段时间，路由器会向所有邻居结点发送它到每个目的节点的距离表，同时它也接收每个邻居结点发来的距离矢量表。这样以此类推，经过一段时间后便可将网络中各路由器所获得的距离矢量信息在各路由器上统一起来，这样各路由器只需要查看这个距离矢量表就可以为不同来源分组找到一条最佳的路由。

现假定用延时作为距离的度量，举一个简单的例子，如图7-37所示。假设某个时候路由器Y收到其邻居路由器X的距离矢量，其中 m 是Y估计到达路由器X的延时。若Y路由器知道它到邻居Z的延时为 n ，那么它可以得知Z通过Y到达X需要花费时间 $m+n$ 。如果Z路由器还有其他相邻路由器，则对于从其他邻居那里收到的距离矢量，该路由器执行同样的计算，最后从中选择费时最少的路由作为Z去往X的最佳路由，然后更新其路由表，并通告给其邻居路由器。

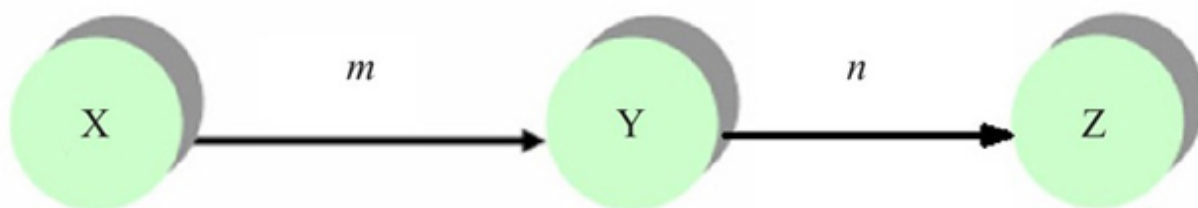


图 7-37 距离矢量路由算法简单实例

现通过图7-38所示的示例介绍距离矢量算法中的路由的确定流程，各段链路的延时均已在图中标注。A、B、C、D、E代表五个路由器，假设路由表的传递方向为： $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E$ （这与路由器启动的先后次序有关）。下面介绍具体的流程。

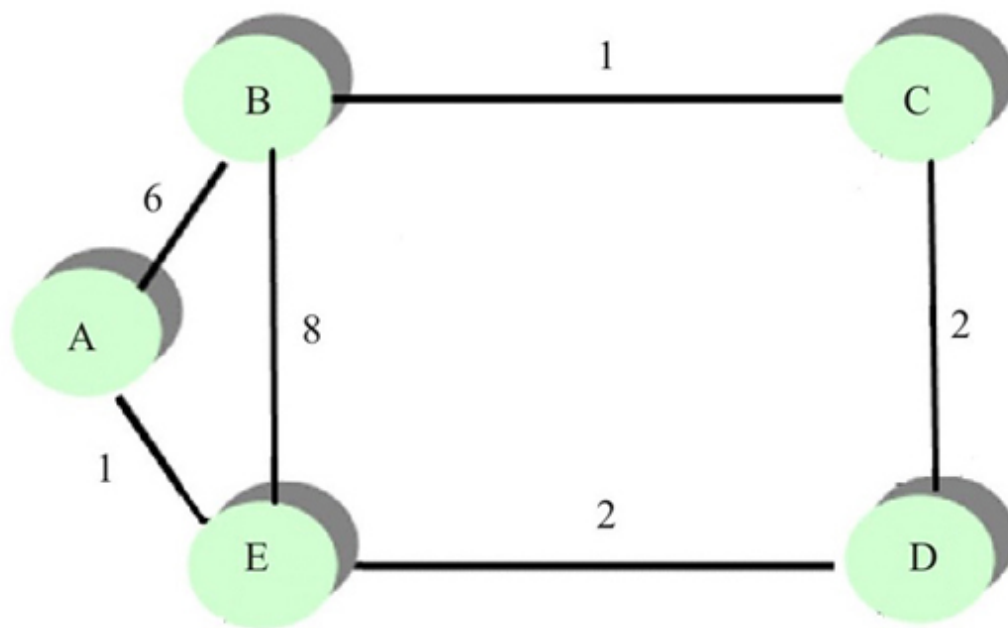


图 7-38 距离矢量算法路由确定示例

1) 初始状态下，各路由器都只收集直接相连的链路的延时信息，各路由器结点得出各自的初始矢量表如图7-39所示。因为各结点间还没有交换路由信息，所以它们的初始状态的路由表同它们的矢量表一样。

A结点初始矢量表	源 \ 目的	B	C	D	E
	A	6			1

B结点初始矢量表	源 \ 目的	A	C	D	E
	B	6	1		8

C结点初始矢量表	源 \ 目的	A	B	D	E
	C		1	2	

D结点初始矢量表	源 \ 目的	A	B	C	E
	D			2	2

E结点初始矢量表	源 \ 目的	A	B	C	D
	E	1	8		

图 7-39 初始状态下各结点的矢量表

2) 现在路由器A把它的路由表发给路由器B。此时路由B会综合从A路由器发来的路由表和自己的初始路由表，更新出一个新的矢量表，如图7-40a所示（最终的矢量表如图中深颜色部分）。从图中可以看出，从B结点到达E结点存在两条路径，一条是直达的，一条是通过A结点到达的。而且这两条线的开销不同，经过A结点到达E结点的开销

(7) 比直达线路的开销 (8) 更低，所以最终在形成的路由表中，把到达E结点的线路改为经A结点这条线路，如图7-40b所示。

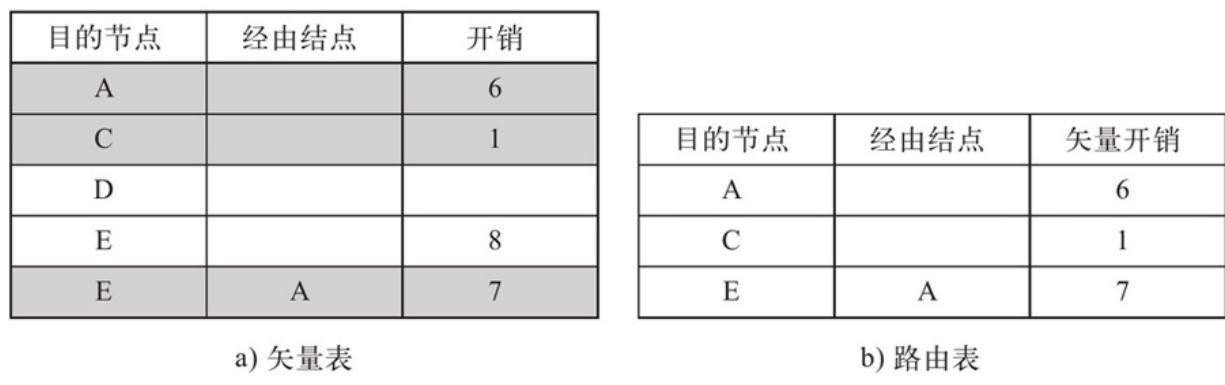


图 7-40 B结点新的矢量表和路由表

3) 路由器B再把最终形成的路由表发给路由器C。同样，路由器C也要把自己原来的初始路由表与从B路由器发来的路由表进行综合，形成新的矢量表，如图7-41a所示（最终的矢量表如图中深颜色部分）。在新的矢量表中，除了最初的直接连接的B和D结点间的矢量外，还新收集了到达A和E结点间的矢量信息。因为C结点没有与A和E结点间的直接连接，在初始路由表中并没有到达这两个结点的路由信息，所以现在只有采用从B路由器发来的路由表，经过B结点到达A、E结点。

这里要注意一点，因为在B结点路由表中已识别了直接通过B结点到达E结点的开销 (8) 还比依次通过B、A结点到达E结点的开销 (7) 大，所以在C结点路由表中是采用依次通过B、A结点到达E结点这条路径。最终形成的路由表如图7-41上图所示。

目的节点	经由结点	开销
A	B	7
B		1
D		2
E	B → A	8

a) 矢量表

目的节点	经由结点	矢量开销
A		7
C		1
D		2
E	B → A	8

b) 路由表

图 7-41 C结点新的矢量表和路由表

4) 路由器C再把它的最最终路由表发给路由器D。同样，路由器D也要把它原来的初始路由表与从C路由器发来的路由表进行综合，形成新的矢量表，如图7-42a所示（最终的矢量表如图中深颜色部分）。在新的矢量表中，除了最初的直接连接的C和E结点间的矢量信息外，还新收集了到达A和B结点的矢量信息。因为D结点没有与A和B结点的直接连接，所以在其最初的路由表中并没有到达这两个结点的矢量信息，此时仍采用经过C结点到达A和B结点的路径。

在这里同样要注意一点，从D结点到达E结点也有两条路径：一是直接到达，二是依次通过C、B、A结点到达，经过比较发现直接连接到达的开销（2）要比通过C、B、A结点到达E结点路径的开销（10）要小，所以在D结点中，到达E结点采用直接连接这条线路。最终形成的路由表如图7-42b所示。

目的节点	经由结点	开销
A	C	9
B	C	3
C		2
E	C → B → A	10
E		2

a) 矢量表

目的节点	经由结点	矢量开销
A	C	9
B	C	3
C		2
E		2

b) 路由表

图 7-42 D结点新的矢量表和路由表

5) 路由器D再把它最终路由表发给路由器E。同样，路由器E也要把它原来的初始路由表与从D路由器发来的路由表进行综合，形成新的矢量表，如图7-43a所示（最终的矢量表如图中深颜色部分）。在新的矢量表中，除了最初的直接连接的A、B和D结点间的矢量外，还新收集了到达C结点的矢量信息，因为E结点没有与C结点的直接连接。此时仍采用经过D结点到达C结点的路径。

在这里有两个要注意的地方：一是从E结点到达A结点的路径问题，因为此时E结点与A结点是直接连接的，而且其开销（1）要比原来从D路由器发来的路由表中提供的通过D、C、B结点到达A结点路径开销（11）要小，所以在最终的E结点路由表中，到达A结点采用直接连接这条线路。二是E结点虽然也与B结点直接连接，但它的开销（8）还要比原来从D路由器发来的路由表中提供的依次经过D、C这两个结点到达B结点的开销（5）大，所以在最终的E结点路由表中，到达B结点

采用依次经过D、C两个结点这条路径。最终形成的路由表如图7-43b所示。

目的节点	经由结点	开销
A		1
B	D → C → B	11
B		8
B	D → C	5
C	D	4
D		2

a) 矢量表

目的节点	经由结点	矢量开销
A		1
B	D → C	5
C	D	4
D		2

b) 路由表

图 7-43 E结点新的矢量表和路由表

通过以上步骤，网络中各路由器就完成了整个路由表的确定，当然在拓扑结构发生变化时，各路由器的路由表又会发生变化，需要重新进行更新。

7.5.4 链路状态路由算法

上节介绍的距离矢量路由算法比较简单，这是它在早期网络几乎无规模可言时能得到广泛应用的主要原因。在网络拓扑结构相对简单且链路极少发生故障时，这种算法的效果完全可以令人满意；然而，对于庞大而复杂的网络，该算法计算新路由的收敛速度极慢，而且在它进行计算过程中，网络处于一种过渡状态，极可能发生循环并造成暂时的拥塞。再者，当前网络底层链路技术多种多样，带宽各不相同，而距离矢量算法对此则视而不见。链路状态路由算法就是为了克服这些缺点才提出并发展起来的。代表协议如OSPF、IS-IS等。

链路状态路由（Link State Routing）算法的基本思想是：网络中各个结点不必交换通往目的站点的距离，而是维护一张网络拓扑图，在网络拓扑结构发生变化时及时更新拓扑图即可。隐藏在链路状态路由算法之后的思想十分简明，它由以下五个步骤组成。

- 1) 发现邻居结点，并知道其网络地址；
- 2) 测量到各邻居结点的延时或开销；
- 3) 组装一个分组，告之刚知道的所有信息；
- 4) 将这个分组发送给所有其他路由器；

5) 使用Dijkstra算法（也就是“最短路径”算法）计算到每个其他路由器的最短路径。

实际上，运行链路状态算法并非一定要采用Dijkstra算法，唯一必须遵从的要求是所有结点都应该使用完全相同的度量制式，也就是说，不论使用什么算法，只要能找出相同的最短路径就行。下面简单介绍以上五个步骤。

1.发现邻居

当一个路由器启动时，它的第一个任务就是找出哪些路由器是它的邻居。为了实现这个目标，它只需要在每一条点到点线路上发送一个特殊的Hello分组即可。线路的另一端的路由器应发送一个应答来说明它是谁，就像我们打电话一样，当一方发出问候时，对方会做出回应，以证明对方就是打电话一方要找的人。路由器的这些名字必须是全局唯一的。

图7-44所示的就是一个由A结点向其邻居结点发送Hello请求分组H_Re，然后邻居结点在收到其Hello分组后发回Hello应答分组H_Rp的示意图。

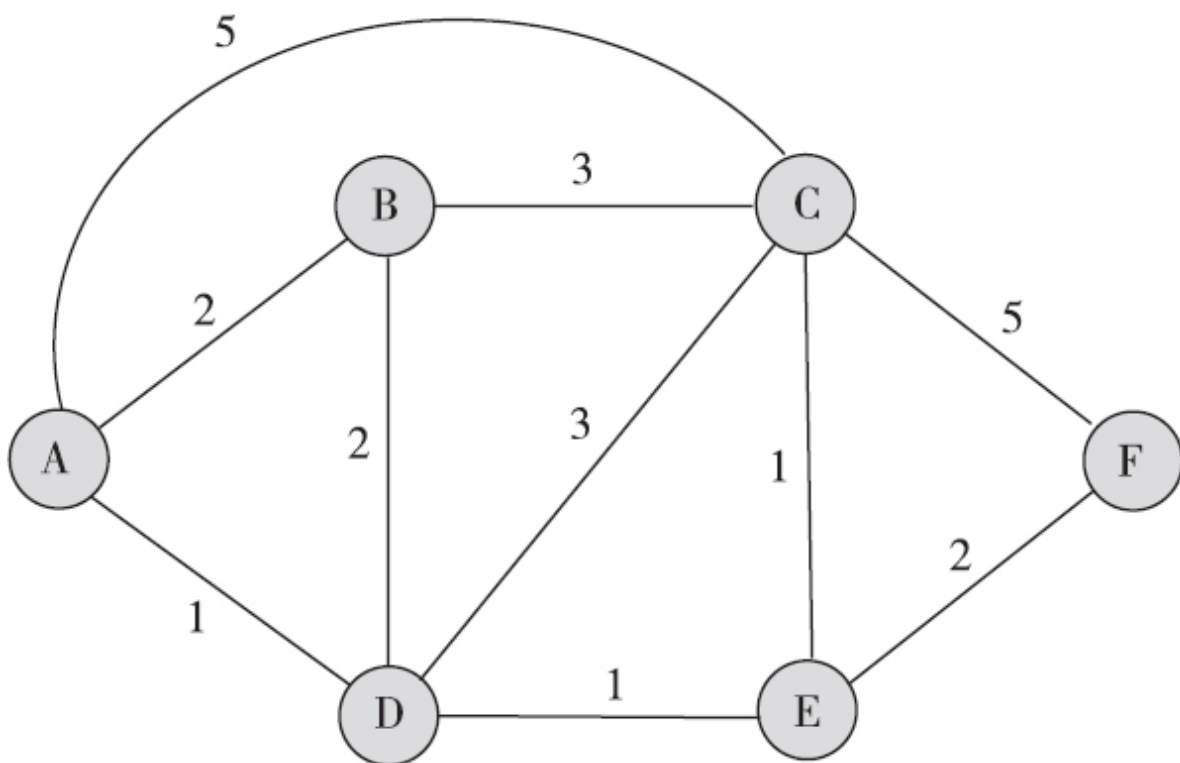


图 7-45 计算链路开销示意图

显然以上我们假设的延时是对称的，也就是分组发送和返回的时间是一样的，但实际上往往并不总是这样。在计算线路延时时要不要考虑线路上的负载，这是一个颇有争议的问题。如果考虑线路上的负载，那么发送时间应从ECHO分组放入队列时算起；如果不考虑线路负载，那么发送时间应从ECHO分组到达队头时算起。考虑线路负载意味着当两条线路的带宽相同时，路由器将选择负载较轻的线路作为最佳输出线路，这种选择能导致良好的性能。但这会使轻负载的线路很快超载，从而导致另一条线路又成为最佳选择，这样容易引起路由表的剧烈振荡，导致不稳定路由和其他潜在问题的产生。只考虑带宽不考

虑负载就不会出现这种问题，但完全不考虑负载也是不合理的，因为它会使流量都集中到高带宽的线路上，而低带宽的线路却很空闲。

以上问题的根源在于路由器总是选择最佳线路转发分组，这很容易引起负载不均衡，因此较明智的做法是将负载按一定比例分布到多条线路上。

3.构造链路状态分组

一旦收集到了各个邻居结点的地址及到各个邻居的线路延时之后，就可以构造一个链路状态分组来携带这些信息。在分组格式中，首先是发送方地址，然后是分组序号和时间（也即链路状态生存时间，TTL），最后是一个邻居列表，每个表项包括一个邻居结点和到达这个邻居结点的总延时，如图7-46所示。

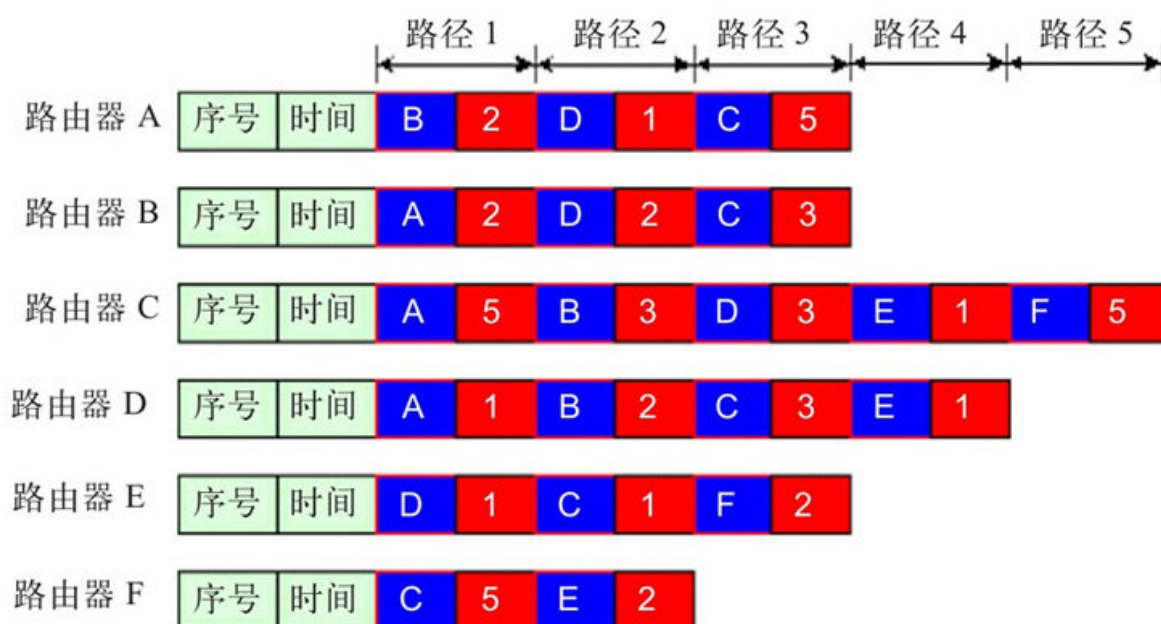


图 7-46 链路状态分组示例

什么时候构造链路状态分组比较合适呢？一种办法是周期性地产生链路状态分组；另一种办法是仅在发现网络拓扑发生改变时才产生，如邻居集合发生了变化，或到邻居的线路延时发生了变化。

4.发送链路状态分组

该算法最复杂的部分是如何可靠地发送链路状态分组。当分组被发送和安装后，首先得到分组的路由器就会用它来改变自己的路由表，此时不同的路由器可能会使用不同的网络拓扑版本，这可能导致不一致、路由环路、不可到达结点和其他问题。

为了可靠地发送链路状态分组，可采用扩散算法（这在前面已有介绍，参见7.5.2节）。每个分组携带一个序号，每当发送一个新的分组时，分组序号就加1。每个路由器维护一张由源路由器、分组序号组成的列表，记录从每个源路由器收到的最新的分组序号。当一个链路状态分组到来时，用分组中的源路由器地址和序号查表，如果这是一个新的链路状态分组，就用扩散算法再将其转发出去，否则将分组丢弃。采用序号来区分新旧分组会有一些问题：一是当序号折回（因为序号也是有限制的，毕竟序号所占的比特位是限定的）时，新的分组都会被丢掉，因为它们与原来接收的分组号；二是当路由器发生崩溃并重启后，只能从序号0开始发送，这些分组也会被丢弃；三是当序号

在传输过程中出错时，如4变成了65540，则其后大量的分组都会被丢掉，因为此时路由器认为对应源路由器上一个最近的分组号就是65540，后面的分组号必须大于这个值，否则就认为是重复接收的。

为解决序号折回的问题，使用了32位长的序号，这样即使每隔一秒发送一个链路状态分组，也需要137年的时间才会发生序号折回，也就彻底解决了折回问题。至于后两个问题，解决的办法是在分组中增加一个寿命域，不管分组在网络中的什么地方，其寿命均每隔一秒减1，当寿命减为0时分组被丢弃。这种方法显然很笨。

还有一些改进的措施可增加算法的健壮性。当一个链路状态分组进入一个路由器并需要扩散时，该分组并不马上放到传输队列里，而是放到一个临时区域中等待一小段时间。如果在该分组被发送前，来自同一个源路由器的另一个链路状态分组到来，则将这两个分组的序号进行比较。如果序号相同，就丢弃后来的分组；如果序号不同，则序号小的分组被丢弃。

为防止分组在路由器到路由器的传输线路上出错，所有链路状态分组都必须被确认。当线路空闲时，临时区域被循环扫描，从中选择一个分组或确认进行发送。路由器将收到的链路状态分组存放在分组缓冲器中。每一行对应一个最近收到的但还未处理完毕的链路状态分组，除记录该分组的源路由器、序号、生存时间、数据以外，对应每

一条输出线路还有发送和确认两个标志位，用来指示是否要在这条线路上转发或确认这个分组。

5.计算新路由

一旦一个路由器收集齐了一整套链路状态分组，它就可以建立完整的通信子网图。在这张图中每条链路均被报告了两次（因为一条链路连接了两个结点，它们会分别进行计算），这两个值可以分开用，也可以求平均值。图建立起来后，就可以运用Dijkstra算法求路由器到任何目的路由器的最短通路了。图7-47所示的就是图7-44所示网络的最终路由表（各路由器上一样）。

路由器 A		路由器 B		路由器 C		路由器 D		路由器 E		路由器 F	
目的地	下一站	目的地	下一站	目的地	下一站	目的地	下一站	目的地	下一站	目的地	下一站
A		A	A	A	D	A	A	A	D	A	E
B	B	B		B	B	B	B	B	D	B	E
C	D	C	C	C		C	E	C	C	C	E
D	D	D	D	D	E	D		D	D	D	E
E	D	E	D	E	E	E	E	E		E	E
F	D	F	D	F	E	F	E	F	F	F	

图 7-47 最终的链路状态路由表示例

7.6 网络拥塞控制方法和原理

网络拥塞现象是指到达通信子网中某一部分的数据包数量过多，使得该部分网络来不及处理，以致引起这部分乃至整个网络性能下降的现象。当拥塞严重时，甚至会导致网络通信业务陷入停顿，即出现死锁现象。这种现象跟公路网中经常所见的交通拥挤一样，在上班高峰时间段中车辆大量增加，在有限的道路宽度下，大量车辆的涌入，使各车辆的行驶速度大受限制，也就出现了拥塞现象，这样每辆车到达目的地的时间都相对增加（即延迟增加），甚至有时还会因拥塞而长时间无法开动，也就出现了局部死锁现象。

之所以会出现拥塞现象，就是因为通信子网中所承受的负荷（即通信子网中正在传输的数据包数）超出了网络的吞吐能力（包数/秒）。当通信子网负荷比较小时，网络的吞吐量随网络负荷的增加而线性增加。当网络负荷增加到某一值后，此时网络的吞吐量达到了最大值。若再加网络负荷，此时的网络吞吐量会不升反降，则表明网络中开始出现拥塞现象了。

就像路上的车一样，当路上车辆数适当时，车辆可以正常行驶。但当路上的车辆数再次增加时，路上的车辆行驶速度会慢下来，此时道路的吞吐量会降下来（此时的“吞吐量”指每秒通过的车辆数）。在一个出现拥塞现象的网络中，到达某个结点的数据包将会遇到无缓冲

区可用的情况，进而出现丢弃现象，从而使这些数据包不得不由源节点重传，使通信子网的有效吞吐量下降。由此引起的恶性循环，使通信子网的局部甚至全部处于死锁状态，最终导致网络有效吞吐量接近为零。

7.6.1 网络拥塞控制方法

为了确保网络的畅通，保持网络的有效吞吐量，就必须对网络中的拥塞现象进行有效控制，同时要尽量避免死锁现象的发生，这就是本节要介绍的拥塞控制（**Congestion Control**）方法。但拥塞控制与流量控制（**Flow Control**）不是完全等同的，拥塞控制需要确保通信子网能够承载用户提交的通信量，是一个全局性问题，涉及主机、路由器，以及与降低网络传输性能有关的所有因素；而流量控制与点到点的通信量有关，主要解决快速发送方与慢速接收方的问题，是局部问题，控制的方法是抑制发送端发送数据的速率，以便使接收端来得及接收，一般都是基于反馈进行控制的，具体可以参见第5章中介绍的流量控制方法。

产生拥塞的原因主要有这三个方面：结点的缓存区容量太小；输出链路的信道带宽不够；处理机的速度不够快。目前通常采用的网络拥塞控制（**Congestion Control**）方法有：缓冲区预分配法、数据包丢弃法和定额控制法三种，下面分别予以介绍。

1.缓冲区预分配法

缓冲区预分配（buffer preallocation）拥塞控制方法常用于虚电路分组交换网中。在虚电路建立时，要求呼叫请求分组所途经的每个结点为此条虚电路预先分配一个或多个数据缓冲区。若某个结点当前无可用的缓冲区（被请求的结点会返回一个“忙”应答信号给呼叫请求分组发送者），则在建立虚电路时，让呼叫请求分组选择其他路径来建立虚电路。这样，通过途经的各个结点就可以为每条虚电路开设永久性的缓冲区（直到虚电路拆除），就总能有空间来接纳并转送经过的数据包。当结点收到一个数据包并将它转发出去之后，该结点向发送结点返回一个确认信息。该确认一方面表示接收节点已正确收到数据包；另一方面告诉发送结点，该结点已空出缓冲区以备接收下一个数据包。

打个比方来说。假设有一批货物要从甲地点运输到乙地点，沿途有A、B、C三个中转站点（相当于虚电路途经的中间结点）可选，如图7-48所示。因为货物较多，为了提高货运周转率，除了要求中转站点有较强的转运能力外，还需要中转站点有较大可用仓储空间（相当于用于解决网络拥塞的缓冲区）可用，以便把那些中转站点来不及转运的货物先保存在仓库中，否则有些来不及转运的货就可能无地可放了。

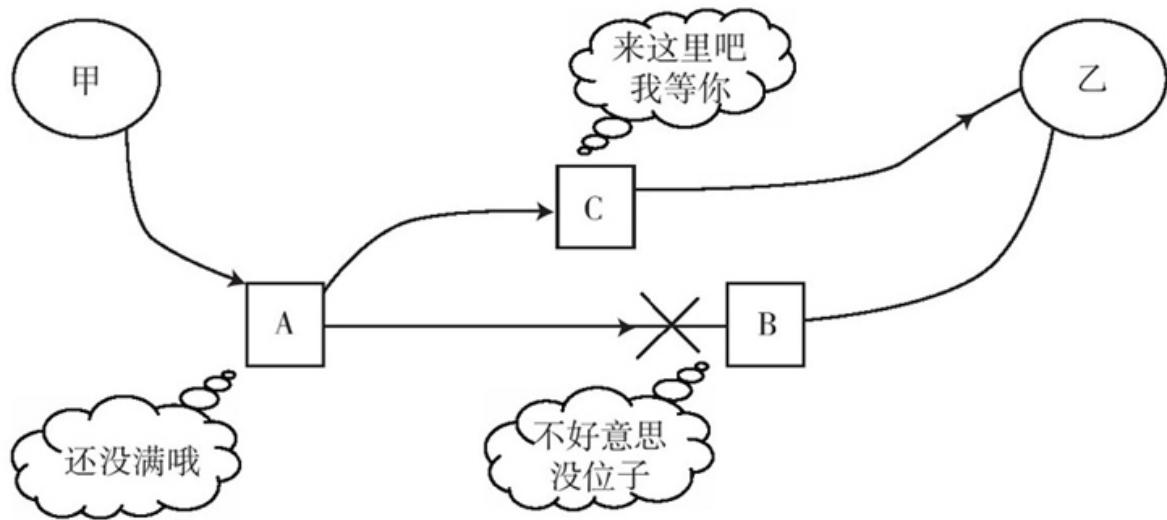


图 7-48 缓冲区预分配法

本来打算走甲-A-B-乙这条运输路线，不过发货方甲通过电话了解到，B站点没有周转用的仓库（相当于中间结点可缓冲区），于是发货方甲打电话问可选的C站点，被告知有仓库可用，于是最终选择了甲-A-C-乙这条货运路径。相当于在建立虚电路时另选其他路径。

当货物正常运输时，每个中转站点都会告诉发货方甲货物转运情况（相当于数据传输时接收方发送的确认分组），这时发货方一方面可以得知哪些货物正常转运了；另一方面也可估算出对应中转站点仓库空间的余量，可以更好地安排发货进度。

以上只是一种同一时刻允许单一数据包处理的情况，若结点之间的协议允许多个未处理的数据包存在，则为了完全消除拥塞的可能性，每个结点要为每条虚电路保留等于窗口大小数量的缓冲区（相当

于图7-48所示的示例中要求中转站点有可以保存多少吨货物的可用仓库空间），而且不管实际上有没有通信量（也就是相当于图7-41中各中转站点的仓库是否可以存满货物）。这样一来，在实际的网络通信中，会有非常多的资源（线路容量或存储空间）被某个连接占有，因此网络资源的有效利用率较低。所以，这种拥塞控制方法主要用于要求高带宽和低延迟的场合。例如传送数字化语音信息的虚电路，是以牺牲资源来换取性能保证的。

2.分组丢弃法

分组丢弃（**Packet Dropping**）拥塞控制方法不必预先为数据包预留缓冲区，而是在路由器的缓冲区已被占满时，只需将后面到来的数据包丢弃，仅在缓冲区腾出空间后再重新接收数据分组。如图7-49所示，从源站点到目的站点间，经过两个路由器，当某个时刻路由器2发现自己的缓冲区满了，不能再接收由路由器1转发来的分组时，就会把后来发来的分组全部丢弃。如图中的packet-n及以后序号的分组。在虚电路服务中，通过计时器控制方法可以使发送被丢弃分组的结点重发这些分组，如图中路由器1重发的packet-n分组。很显然，这是一种最直接，但同时也是一种最粗的拥塞控制方法，因为其直接拒收后面到来的数据包。

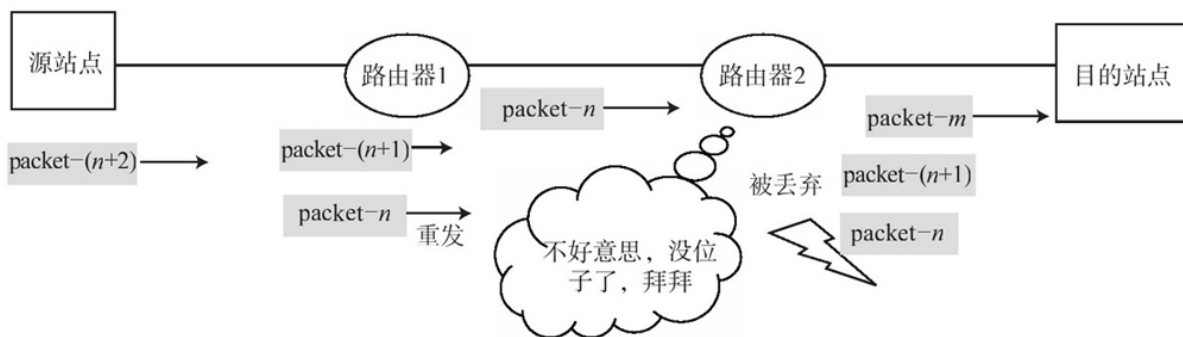


图 7-49 “分组丢弃”拥塞控制方法基本原理

从这种拥塞控制方法的控制原理可以看出，如果通信子网提供的是数据报服务，采用这种拥塞控制方法来防止拥塞的发生则不会引发大的影响，仍可按前面介绍的“缓冲区预分配”拥塞控制方法来解决，选择其它路径重新发送，如图7-48所示。因为在数据报服务的分组报文中，每个分组都有完整的路由地址信息，可以独立地选择最佳路径进行路由，当发现原来的路径性能不佳时，会自动重新以其他更好的路径进行分组交换；另外，在数据报服务中，接收结点在接收到分组后不需要进行确认，所以即使接收结点把数据包丢弃了，也不会引起发送结点长时间等待确认应答。

如果通信子网提供的是虚电路服务，则必须在某个前面的结点中保存了被丢弃数据包的备份，以便拥塞解决后能重新传送。因为虚电路服务的分组报文中，没有独立的路由地址信息，也不能单独地被路由，同一数据包的所有分组必须走同一条虚电路，而且要求接收结点在收到每个分组后必须有确认应答。如果虚电路中某个数据包的某个

分组报文被丢弃了，发送结点就不会收到对应分组的应答确认。这里有两种解决被丢弃数据包重发的方法：一种是就是采用计时器方法，当某个分组发送后，在计时器超时后仍没有收到接收结点发回的对应分组确认应答，就要使某个转发该数据分组的结点重新发送对应的数据分组，直至数据分组被收到，毕竟每一结点的缓冲区可用空间是在不断动态变化的，因为它们同时也在不断进行转发数据分组；另一种方法是让发送被丢弃数据包的结点在尝试重发一定次数后放弃发送，并迫使数据源结点超时而重新开始发送。

3.定额控制法

定额控制拥塞控制方法是在通过通信子网中设置适当数量称为许可证的特殊信息来实现的。其中一部分许可证在通信子网开始工作前预先以某种策略分配给各个源节点；另一部分则在子网开始工作后在网中四处环游。当源节点要发送来自源端系统的数据包时，它必须首先拥有许可证，并且每发送一个数据包注销一张许可证。目的节点则每收到一个数据包，并将其递交给目的端系统后，便生成一张许可证。这样便可确保子网中数据包数不会超过许可证的数量，从而防止拥塞的发生。但这时网络的传输效率可能不能发挥到最佳。

其实这种控制方法有点像令牌网上的介质争用解决方案。在令牌网中，源节点要发送数据必须要获取网络中唯一的令牌，不同的是这里的证书不是唯一的，而是有许多。

7.6.2 死锁及其预防

死锁就是一些结点由于没有空闲缓冲区而无法接收和转发数据包（转发时也需要一定的缓冲空间的）的现象，死锁的结果是结点间相互等待，既不能接收数据包也不能转发数据包，并一直保持这一僵局，严重时甚至导致整个网络的瘫痪。死锁是拥塞的极端后果，是网络中最容易发生的故障之一，即使在网络负荷不很重时。此时，只能靠人工干预来重新启动网络（类似于因死机而进行的系统重启）解除死锁，但重新启动后并不能消除引起死锁的隐患，所以可能再次发生死锁。下面介绍两种常见死锁情况及预防方法。

(1) 存储转发死锁及其防止

最常见的死锁是发生在两个结点之间的直接存储转发死锁。例如，A、B和C三个结点的缓冲区都满了，不能再接受更多的分组。A的分组都是发往B的，而B由于缓冲区满了而不能接收任何分组。这样，A要等到B发送掉一些分组，释放出缓冲空间后才能发送。B的分组是送往C的，而C的缓冲区也满了。C的分组是送往A的，而A的缓冲区同样满了。换句话说，A等待B清空缓冲区，B等待C清空缓冲区，而C则等待A清空缓冲区，但事实上当前情况下，这三个结点的缓冲区

都已无法清空了。当一个结点处于死锁状态时，所有与之相连的链路将完全被拥塞。

一种防止存储处于死锁的方法是为每个结点设置 $m+1$ 个缓冲区（ m 为通信子网的直径，即从任一源结点到任一目的结点间的最大链路段数），并顺序编号。每个源结点仅当其为0号缓冲区可用时才能接收源端系统来的数据包，而此数据包仅能转发给1号缓冲区空闲的相邻结点，再由该结点将数据包转发给它的2号缓冲区空闲的相邻结点，以此类推。最后，该数据包或顺利到达目的结点，并被递交给目的端系统；或到了某个结点编号为 m 的缓冲区中，再也转发不下去，此时一定发生了循环，应该将该数据包丢弃。由于每个数据包都是按照编号递增规则分配缓冲区的，所以结点之间不会相互等待空闲缓冲区而发生死锁现象。这种方法的不足之处在于当某结点虽然有空闲缓冲区，但正巧没有所需要的特性编号的缓冲区时，数据包仍要等待，从而造成了缓冲区和链路的浪费。

另一种防止存储转发死锁的方法是使每个数据包上都携带一个全局性的唯一的“时间戳”，每个结点要为每条链路保留一个特殊的接收缓冲区，而其他缓冲区均可用于存放中转数据包，按每条输出链路的队列上数据包时间戳顺序排队。例如，结点A要将数据包送到结点B，若B结点没有空闲缓冲区，但正巧有要送到A结点的数据包，此时A、B结点可通过特殊的接收缓冲区交换数据包；若B结点既没有空闲缓冲

区，也没有要送到A结点的数据包，B结点只好强行将一个出路方向大致与A结点方向相同的数据包与A结点数据包互相交换，但此时A结点中的数据包必须比B结点中的数据包具有更早的时间戳，这样才能保证子网中某个最早的数据包不受阻挡地转发到目的地。由此可见，每个数据包最终总会一步一步地发送到目的结点，以此来避免死锁现象的发生。

(2) 重装死锁及其防止

重装死锁就是在数据分组重新组装、恢复成原来数据包的过程中发生的死锁，是另一种比较严重的死锁现象。假设源结点发送的报文很长，被源结点拆成若干个数据分组发送，当这些分组到达目的结点后，就要将这些具有相同报文序号的多个分组重新组装成一个完整的报文再递交给目的端系统。若目的结点用于重装报文的缓冲区空间有限，而且它无法知道正在接收的报文究竟被拆成多少个数据分组，此时就可能发生死锁现象，因为它没办法完成整个分组的重装，在缓冲的分组就不能被发送出去。而此时相邻结点仍在不断地向它传送数据分组，因没有可用的缓冲空间，本结点就无法继续接收这些后续的分组了。虽然相邻结点可以重发，但目的节点的缓冲区始终得不到清空，这样源节点在经过多次尝试后，相邻结点就会绕道从其他途径再向该目的端系统传送数据分组，但这样做又会造成周边区域也由此发生拥塞。

以下几种方法可以避免重装死锁的发生。

□允许目的结点将不完整的报文递交给目的端系统，只要按顺序发送即可，完整组装的任务交给目的端系统。

□检测出不能完整重装的报文，然后要求发送该报文的源端系统重新传送，但这种方法的有效性比较低，因为这也很难确保目的节点在重发时就有足够的缓冲空间。

□每个结点配备一个专门的缓冲空间，用以暂存不完整的报文，相当于CPU上配置的多级缓存，但这明显会增加设备的成本。

以上前三种方法中，第1种方法可能使端系统接收到不完整的数据报，而使整个数据无效。第2种方法使端系统中的协议复杂化了，同时加重了源端负担。第3种方法虽然不涉及端系统，但却使每个结点增加了开销，系统性能会大受影响。

7.7 网络层设备及主要技术

在网络层中，目前最主要的设备就是路由器和三层交换机。路由器为了实现不同网络的互连，又包括了许多软、硬件技术，如前面介绍的各种路由技术，还有由于路由器是位于网络边界的设备，作为网络的唯一出口和入口，所以它的工作负荷也相当重，这也使得路由器在硬件性能方面有它的特殊性所在。主要的路由技术在前面已有介绍，各主要的路由协议将在下章再具体介绍，下面先来了解一下路由器的一些主要硬件技术。

7.7.1 路由器主要硬件技术

路由器的硬件技术主要体现在以下几个方面。

1. 硬件体系结构

路由器的硬件体系结构大致经历了6次变化，从最早期的单总线、单CPU结构发展到单总线、多CPU再到多总线多CPU。到现在，高速IP路由器中多借鉴ATM的方法，采用交叉开关方式实现各端口之间的线速无阻塞互连。高速交叉开关的技术已经十分成熟，在ATM和高速并行计算机中早已得到广泛应用。伴随着高速交叉开关的引入，一些相应的技术问题也同时被引入了，特别是针对IP多播、广播以及服务

质量（QoS），引入了一些成熟的调度策略和算法技术，使这些问题都得到了很好的解决。

为了缓解互连瓶颈，在最近几年涌现出许多新的系统和解决方案，其中采用ASIC（专用集成电路）来完成规定的数据包处理工作是十分理想的。但它的开发周期太长，复杂的ASIC要18个月到2年时间，每一个ASIC的开发都必须经历一个设计和制造的周期，适应不了当今越来越短的产品开发周期。

在新一代路由器中，在关键的IP业务流程处理上采用了可编程的、专为IP网络设计的网络处理器技术。它通过若干微处理器和一些硬件协处理器并行处理，通过软件来控制处理流程。对于一些复杂的操作（如内存操作、路由表查找算法、QoS的拥塞控制算法、流量调度算法等）采用硬件协处理器来提高处理性能。

网络处理器位于每个线卡上，采用软件完成IP报文处理和转发，处理器的软件可以更新，不仅满足了业务发展的需要，而且可提供线速转发的性能。网络处理器从2000年年初出现到现在被许多网络设备制造商选作新一代高端路由器设备的核心处理器。而在这段时间里，能够开发出成熟的NPU芯片的公司也从开始的两三个迅速增加到了十几个，而且NPU的处理能力也从2.5Gbps扩展到10Gbps。这些都说明网络处理器技术在网络产品市场中的位置越来越重要。尤其在高端路由

器市场，网络处理器以其杰出的包处理性能及可编程性已经成为构成路由转发引擎不可替代的部分。

2.ASIC技术

在路由器中，要极大地提高速度，首先想到的是ASIC，ASIC可以用作包转发、查路由，并且目前已经有专门用来查找IPv4路由的商用ASIC芯片。ASIC技术的应用使路由器内的包转发速度和路由表项查找速度有了显著的提高。

高速路由器将路由计算、控制等非实时任务同数据转发等实时任务分开，由不同部分完成。路由计算、控制等非实时任务由CPU运行软件来完成，数据转发等实时任务由专门的ASIC硬件来完成。自1997年下半年以来，一些公司开始陆续推出采用ASIC进行路由识别、计算和转发的新型路由器，而转发器负责全部数据转发功能。

ASIC技术的进展意味着更多的功能可移向硬件，提高了性能水平，增加了功能。与软件执行相比，ASIC的性能是其3倍。但是全硬件化的路由器使用起来缺乏灵活性，且会冒一定的风险，因为标准规范仍在不断演变过程中，于是出现了可编程ASIC。可编程ASIC是ASIC的发展趋势，因为它可通过改写微码来适应网络结构和协议的变化。目前，有两种类型的可编程ASIC：一种以3Com公司的FIRE（Flexible Intelligent Routing Engine）芯片为代表；另一种以Vertex

Networks的HISC专用芯片为代表，HISC芯片是一颗专门为通信协议处理而设计的CPU，通过改写微码，使芯片具有处理不同协议的能力。

3.分布式处理技术

最初的路由器采用了传统计算机体系结构，包括共享中央总线、中央CPU、内存及挂在共享总线上的多个网络物理接口。接口卡通过总线将报文传送到CPU，CPU完成路由计算、查表、做转发决定等处理，然后又经总线送到另一个物理接口并发送出去。这种单总线单CPU的主要局限是处理速度慢，一颗CPU完成所有的任务，从而限制了系统的吞吐量。另外，系统容错性也不好，CPU若出现故障容易导致系统完全瘫痪。这一切都致使传统路由器的转发性能很难有大的提高。

现代的路由器对报文转发采用分布式处理，可以插多个线路处理板，每个线路板独立完成转发处理工作，即做到在每个接口处都有一个独立的CPU，专门负责接收和发送本接口数据包，管理接收发送队列、查询路由表并做出转发决定等。通过核心交换板实现板间无阻塞交换，即一个板上输入的报文经过寻路后可以像通过导线直连那样，被交换到另一个板上输出，实现包交换，其整机吞吐量可以成倍扩充。而主控CPU仅完成路由器配置控制管理等非实时功能。这种体系结构的优点是本地转发/过滤数据包的决定由每个接口处理的专用CPU来完成，对数据包的处理被分散到每块接口卡上。线路板上有专用芯

片完成二层、三层乃至四层的转发处理工作，硬件实现使转发能够达到线速（高速端口所连接线路的速率），具有了电路交换同样的性能，使路由器不会成为网络中的瓶颈。

然而，单总线结构路由器存在一个最大缺陷：一次只能有一个分组从入口交换到出口。如果能在入口和出口之间有多条数据传输通路，则能解决这种问题，同时可大大提高系统的吞吐率。基于这种想法，同时借鉴ATM交换机结构的优点，提出了基于交换机结构的新一代路由器体系结构，具体可参见前面的介绍。

4.交换式结构和调度算法

交换结构有Cross bar（矩阵）、共享存储器和总线三种方式。矩阵结构的速度由调度器决定，共享存储器结构的速度由存储器的读写速度决定，共享总线结构的速度由总线的容量和仲裁的开销决定。

调度器是矩阵结构的核心，它在每个调度时隙内收集各输入端口有关数据包队列的信息，经过一定的调度算法得到输入端口和输出端口之间的一个匹配，提供输入端口到输出端口的通路。采用输入缓冲无阻塞方式的矩阵，用ESLIP（Extended SLIP，扩展Internet串行线路协议）算法实现调度已被一些厂家所采用。调度器设计的难点在于，既要满足系统吞吐率达到100%的要求，又要支持CoS（Classes of Service，服务级别）。调度算法中，加权公平排队算法（WFQ-

Weighted Fair Queuing) 和经过改进后的加权公平流排队算法 (WF2Q-Weighted Fair Flow Queuing) 比较容易实现, 而且性能也不错。总之, 要让互联网真正能够综合多种业务, 作为网络连接核心设备的路由器, 必须提高端口速率和交换容量, 提供QoS保证和流量工程 (TE-Traffic Engineering) 功能。

在交换结构中, 存在单级交换结构和多级交换结构两种, 下面分别介绍。

(1) 单级交换结构

单级交换结构的引入逐步克服了共享总线的以上缺点。从技术上, 目前使用较多的单级交换结构有共享内存和矩阵交换两种。矩阵交换的结构由于其简单性得到了更多的青睐和更广泛的采用。共享内存结构是通过共享输入输出端口的缓冲器, 从而减少了对总存储空间的需求。矩阵结构可以同时提供多个数据通路。一个矩阵交换结构由 $N \times N$ 交叉矩阵构成。当交叉点 (X, Y) 闭合时, 数据就从X输入端输出到Y输出端。交叉点的打开与闭合是由调度器来控制的。因此, 矩阵结构的速度取决于调度器的速度。调度器是矩阵交换结构的核心, 它在每个调度时隙内收集各输入端口有关数据包队列的信息, 经过一定的调度算法得到输入端口和输出端口之间的一个匹配, 提供输入端口到输出端口的通路。

矩阵交换结构可以支持高带宽的原因主要有两个：首先，线路卡到交换结构的物理连接已简化为点到点的连接，这使得该连接可以运行在非常高的速率上。半导体厂商目前已经可以用传统CMOS技术制造出1Gbps的点对点串行收发芯片，并且可以在今后几年里把速度进一步提高到4~10Gbps的水平。其次是它的结构可以支持多个连接的同时能以最大速度传输数据，这一点极大地提高了整个系统的吞吐量。只要同时闭合多个交叉节点，多个不同的端口就可以同时传输数据。从这个意义上来说，我们认为所有的矩阵在内部是无阻塞的，因为它可以支持所有端口同时以最大速率传输（或称为交换）数据。

数据包通过矩阵的时候，可以是定长单元的形式（通过数据包的定长分割），也可以不进行分割直接进行变长交换。一般高性能的矩阵交换结构都采用了定长交换的方式，在数据包进入矩阵以前把它分割为固定长度的cells，等这些cells通过交换结构以后再按照原样把它组织成原来的变长包。

共享内存的特点是实现简单，能达到比较高的吞吐率，但是其可扩展性比较差，当线路接口卡数量较多时，性能将受到一定的影响。而与之对应的交叉开关能够达到比较高的速率，扩展性好，但是需要设计完善的调度算法并用高速硬件实现调度器。随着人们对交叉开关调度算法研究的深入，已经设计并实现了许多性能良好、实现简单的

调度算法。因此，目前高性能路由器都趋向于使用交叉开关作为交换结构。交叉开关和共享内存结构都属于单级交换结构范畴。

当考虑大型系统时，单级交换结构有两个基本问题。第一、对于小规模系统，端口成本还算合理，但随着规模的扩大，其成本涨得也非常快。第二、所有的单级交换结构在技术上受限于其尺寸与速度。一旦达到这些极限，单级交换机无法再增加端口或提升线路速率。正因为如此，可扩展的交换系统必须采用多级结构。

(2) 多级交换结构

多级交换结构是由多个交换单元互连起来的，每个交换单元具有一整套输入输出，与普通交换机类似，提供输入输出的连接。通过互连多个小的交换单元，就可以制造一个大型的、可扩展的交换结构。多级结构之间的不同取决于交换单元之间是如何互连的。典型的结构包括Benes网、Butterfly网、Clos网等形式。

Benes网使用方形交换单元（输入输出端口数相同）进行多级互连。一般来说，3级N部Benes网的每一级均可以用N个输入/输出端口和N个交换单元来构造。这个格式结构在输入端和输出端之间形成N个可能的通路。Benes输出可以扩展至任意级数。

虽然对于小型系统来说单级结构的设计相对简单，成本也相对较低，但是它不能满足下一代Internet扩展的需要。多级结构在操作上虽

然较复杂，但是可以扩展到成百上千个端口，这对于下一代Internet核心路由系统是绝对必要的。在多级拓扑结构中，**Benes**结构是最佳选择，因为它的系统复杂程度最低，性能好且满足可扩展的要求。

5.路由表硬件查找技术

传统的基于软件的路由查找策略，如树或哈希算法，其执行过程都是相当慢的，而且与路由表的大小相关联。在路由表更新时，输入的数据包必须被缓存或丢弃，降低了路由器的性能。

为了解决地址资源紧缺，减少路由表的规模，降低管理难度，互联网采用了CIDR（Classless Inter-Domain Routing，无类域间路由）。这样，路由表中存放的不是一个个具体的IP地址，而是可变长度的网络前缀。路由器在对IP包寻址时，采用最长的网络前缀匹配（LPM-Longest Prefix Matching）。例如，假设路由表中有两个表项“202.168.X.X，输出端口1”和“202.168.16.X，输出端口2”（X表示任意），如果有一个IP包的目的地为202.168.16.5，那么这个包应该从端口2输出。传统的路由器执行最长网络前缀匹配的时间很长，使得路由表查找成为路由器速度的瓶颈。近两年出现了一些快速查表算法，能够支持吉比特链路。这些算法包括改进的精确匹配法、基于trie（线索）树法、并行硬件法、协议改变简化路由表的查询和缓冲算法等。这些算法，有些适合硬件实现，有些适合软件实现。对于组播地址寻

址，要根据IP包的源地址和组播地址查表，对源地址采用最长前缀匹配法，对目的地址采用精确匹配法。

基于软件的查找和更新路由表的不确定性增加了包传输时的抖动，因此必须进行包的缓存，另外其在高速率时还会造成丢包。所以，这些方法只能用于比较小的、性能较低的包转发应用。为了适应网络的发展，理想的包转发方案必须不但能够保证线速的数据转发速率，并且能够提供足够大的路由表来满足下一代的路由设备的需要（在边界位置应达到512kB）。同时它还要能够以很小的更新时延来处理长时间的突发路由表更新。尽管通常路由表的更新为每秒几百次，但瞬间突发更新则可能会高出很多。要解决这个问题，目前来看最为有效的办法是采用专门的协处理器结合内容寻址寄存器（CAM）来完成快速路由查找或更新，即硬件实现路由表的查找。

6.高密度和多端口

路由器的高速接口是建设大容量IP光网的基础，具有高速OC-192光接口的高速路由器是网络向IPoverD WDM结构发展的关键。在新的阶段，作为互联网的核心动力引擎，电信运营商不仅对高端路由器的处理速度不断提出新的要求，尤为重要，他们开始关注其业务功能。运营商不仅需要构建速度快的网络，更需要可赢利、利于开展业务的核心网。业务实现技术是高端路由器是否符合电信企业实际运营

要求的关键，这既包括丰富的连接端口技术，又包括对增值服务的支持功能。

由于运营商的网络基础不同，应用场合也千差万别，因此要求高端路由器有强大的适应性和灵活性，有对各种速率线路卡的支持能力，从DS3（一种最高传输速率达45Mbps的网络连接技术）直到10Gbps，并且支持快速以太网/吉比特以太网、动态分组传输、ATM（异步传输模式）等技术。它采用的高密度方案能够集合大量业务，包括基于ATM的话音业务、DSL高速接入、专线集合和互联网服务供应商集合，可以应用在广域网骨干、城域网骨干、城域网边缘、园区网骨干等多种网络环境。对用户来说，高的端口密度具有重要的意义，因为中心局和入网点的空间资源不但有限而且成本很高。在巨大的压力下，业务提供者已经开始考虑如何节约空间，而具有高密度端口的高速路由器是解决空间压力的一个重要方面。较低的高速端口密度对于今后的竞争是一个不利的因素，目前各路由器厂商希望通过引入新的输入输出模块，解决端口密度问题。

7.光路由器技术

新一代Internet技术只需部署IP设备和必要的光设备、高速路由器就可直接进入国家骨干网，成为国家核心网。这意味着，IP网的协议能够与光网络很好地结合，由原来的SDH/SONET到主要支持IP。为了达到这一点，核心路由器应具备新的光接口、新的网络控制协议等。

IP over Optical是简化IP骨干网的良好解决方案，可以去除昂贵的SONET和ATM设备，而且极大地降低了网络管理的复杂性。但是高速路由器直接连接到DWDM系统时，网络只有静态的点到点通路，不允许业务在不同光通路间进行交换，这导致了较差的灵活性。为解决此问题，国外很多大的运营商和设备商提出了将光交换机作为高速路由器和DWDM系统的中间层的概念，进一步将选路和光交换平台综合成光路由器。使用IP协议、通过在不同波长间交换业务、允许动态控制带宽等，为开展新业务提供了更多的灵活性。

这方面的技术进展完全源于光纤通信技术的进展。随着IP核心地位逐渐被认同，IP over ATM和ATM over SDH方式直接被IP over SDH方式取代。SDH采用时分复用的方式承载多路数据。因此在核心网中需大量采用复用器交叉连接器，DWDM（密集波分复用）使得一根光纤上可用不同的波长传送多路信号。

光路由器在网络核心各光波长通道之间设置MPLS协议和波长选路协议（WaRP）控制下的波长选择器件，实现选路交换，快速形成新的光路径。波长的选路路由由内部交叉矩阵决定，一个 $N \times N$ 的交叉矩阵可以同时建立 $N \times N$ 条路径，波长变换交叉连接可将任意光纤上的任意波长交叉连接到使用不同波长的任意光纤上，具有很高的灵活性。

目前，国内外的电信设备供应商（TEP）和IP设备供应商（IEP）都在加紧研制开发系列化的光交换/光路由产品。光路由器产品主要有

Cisco的ONS15900光路由器， Corvis的CoreWave光路由器， Monterey Networks公司的Monterey 20000光路由器。

7.7.2 路由器主要软件技术

在软件方面同样具有许多复杂的路由器技术，具体包括以下几个方面。

1.VPN技术

VPN（Virtual Private Network，虚拟专用网）是路由器的重要技术之一。VPN是指在公用网络上建立虚拟私有网的一种新兴网络技术，目前随着技术的成熟，在各大企业中应用比较普遍。VPN通信根据不同的实现方式可以分为如下几种类型。

按接入方式划分，VPN可以分为专线VPN和拨号VPN两类。专线VPN是为已经通过专线接入ISP边缘路由器的用户提供的VPN实现方案，是属于Site-to-Site（端到端）的网络与网络之间互联的VPN方式，在微软的Windows系统中称为路由器到路由器VPN。这类VPN通信适用于网络用户之间，它的发起端可以是双方的任意一方。而拨号VPN（又称VPDN）是指利用拨号PSTN或ISDN接入向企业网络以拨号方式进行的VPN通信，它属于Client-to-Site（客户到站点）方式，在微软的Windows系统中称为远程访问VPN。这类VPN通信适用于移动用户、外出办公用户，它的发起端必定是拨号用户端。

按协议类型划分，VPN又可以分为第二层隧道协议VPN和第三层隧道协议VPN两类。二层隧道协议VPN就是指所进行的VPN通信是利用二层隧道协议进行的。二层隧道协议又有点到点隧道协议

（PPTP）、第二层转发协议（L2F）和第二层隧道协议（L2TP）三种，凡是采用这三种隧道协议之一的VPN通信均为第二层隧道协议VPN。第三层隧道协议VPN是指利用第三层隧道协议进行的VPN通信，第三层隧道协议又有通用路由封装协议（GRE）和IP安全

（IPSec）协议两种，最常用的是IPSec安全协议。目前的VPN通信基本上是第二层隧道协议与第三层隧道协议的有机结合，如L2TP+IPSec协议的VPN通信。

路由器是VPN通信中非常重要和关键的设备，绝大多数企业网络之间的VPN通信是通过路由器发起的。当然在整个VPN通信市场中，除了采用路由器外，还有采用交换机、集中器和远程访问服务器进行的，这些VPN方案均有它们各自不同的技术特。

2.QoS技术

QoS（Quality of Service，服务质量）原先是ATM网络中中专用的，但利用IP传输VOD（Video On Demand，视频点播）等多媒体信息的应用越来越多，IP作为一个打包的协议显得有点力不从心，这主要体现在：延迟长且不为定值，丢包造成信号不连续且失真大。为解决这些问题，厂商提供了若干解决方案：第一种方案是基于不同对象

的优先级，某些设备（多为多媒体应用）发送的数据包可以后到先传；第二种方案基于协议的优先级，用户可定义哪种协议优先级高，可后到先传，Intel和Cisco都支持；第三种方案是进行链路整合MLPPP（Multi Link Point to Point Protocol，多链路点对点协议），Cisco支持将连接两点的多条线路进行带宽汇聚，从而提高带宽；第四种方案是进行资源预留RSVP（Resource Reservation Protocol，资源预留协议），它将一部分带宽固定的分给多媒体信号，其他协议无论如何拥挤，也不得占用这部分带宽。这几种解决方案都能有效提高传输质量。

下面简单介绍与QoS有关的路由器技术。

（1）数据包分类技术

IPv4包头中有一个3位的区域用以标识此IP包的优先级。据此优先级，IP路由器可以决定不同IP包的转发优先顺序。可以说，自IP协议制定之日起，就为日后提供更好的QoS做了准备。

路由器对到达的分组包进行识别、分类以决定其所应接受的服务类型。当初IETF所考虑的方案是在网络的核心，根据IP报头的ToS

（Type of Service，类型服务）域来识别分组，但是在互联网的发展过程中，由于一直采用“尽力”传输模式，同时由于终端在发送IP包时不考虑ToS，因此，ToS一直没有发挥作用。目前在边缘设备，根据IP分

组的源IP地址、目的IP地址、源端口号、目的端口号、传输层协议类型来对分组进行识别。此外，为了实现防火墙的功能也需要对IP分组进行识别。

在识别时，每条识别规则采用的是源IP地址、目的IP地址、源端口号、目的端口号、传输层协议类型。在上述识别规则中，每个域都可能是一个区间。例如有这样一条识别规则“126.66.64.X, 126.66.72.X, X, 23, TCP”（X表示任意），这条规则识别从网络126.66.64.X到网络126.66.72.X的telnet数据。从几何的角度来看，假如判别时利用了IP报头的K个域，则这个问题实际上是在一个K维空间中许多互相交叠的实体（每条判别规则对应于一个实体），每当有一个分组到达时，该分组相当于K维空间上的一个点，进行判别实际上是要找出包含该点的优先级最高的实体。

（2）数据流的分类

一系列通过给定的源和目的地的数据包被视为数据流，数据流可以是长时间维持的TCP连接的一系列数据包，也可以是声音或图像的一系列UDP数据包。通常，流有长短之分，划分有两个标准：端口对和主机对。按端口对划分是指同一流的数据包必须具有相同的源、目的地址和TCP/UDP端口号等；按主机对划分只要求各数据包具有相同的源和目的地址。目前常用的流分类器有三种，一种是X/Y分类器，Y为规定的时间间隔，X为数据包数。若在时间Y内某一数据流到达的数

据包数大于X，则该流就被认定为长数据流，否则是短数据流；另一种是协议分类器（Protocol Classifier），它规定了所有的TCP包均被定义为数据流；还有一种是端口分类器（Port Classifier），它规定了几个特殊的TCP端口作为长数据流。因为分类要对每个进入路由器的包进行包头检查，需要快速的分类算法。

（3）RSVP（资源预留协议）技术

这是IP路由器为提供更好的服务质量向前迈进的具有深刻意义的一步。传统IP路由器只负责包转发，通过路由协议知道邻近路由器的地址。而RSVP则类似于电路交换系统的信令协议，为一个数据流通知其所经过的每个结点（IP路由器），与端点协商为此数据流提供质量保证。

（4）DiffServ技术

近年来，IETF又新推出另一种QoS策略——DiffServ（Differentiated Service，差别服务）。目前DiffServ的框架已基本确定，美国的Internet2也选择DiffServ作为其QoS策略。与DiffServ相比，RSVP是一种Integrated Service（集中控制策略），而DiffServ则是一种分散控制策略，其精髓是仅控制路径中的每一行为。终端应用设备通过SLA（Service Level Agreement，服务级别协定）与边缘路由器协商

获得其应用数据流可得到保证的服务级别。除此之外，多协议标记交换（MPLS）也常被用来解决QoS问题。

3.MPLS技术

IP网络的发展存在着一个非常明显的障碍，这是由IP协议本身固有的一个缺陷决定的，那就是它是一个无连接的协议，因此IP网络上的应用无法得到很好的QoS保证。由于缺乏连接性，每一个IP包都是单独发到目的地的，网络中的各个结点都无从知晓这些无连接的包中的某一个是如何到来的。与此相比，面向连接的协议，如帧中继则需要建立一个固定的虚电路。连接路径上的各个结点以及干线可以先为其预留资源，以提供QoS保证。但也是因为它的无连接性，IP协议却具有其他网络协议所无法比拟的灵活性，这一点通过Internet已经得到了证明。如果能结合这两类协议则肯定是非常有意义的，而且结合后的协议还能具有两类协议的优点，那应用效果会更好。就是在这样一种思想下产生了MPLS（Multi Protocol Label Switch，多协议标记交换）协议。MPLS结合的是IP路由和标记交换这两种技术，IP路由当然是IP协议的功能了，而标记交换原先则是ATM这类有连接协议的功能。

MPLS在网络入口的边缘路由器（LER）为每个包加上一个固定长度的标签，核心路由器根据进入包头的标记值进行标记交换，作出本地转发决定，在出口的边缘路由器再恢复原来的IP包。从入口到出口

的一连串这样的标记交换和转发决定操作，形成了一条标记交换路径。**MPLS**是处于第二层和第三层之间的一种技术，它紧密地将第二层和第三层结合在一起，充分发挥了第二层的交换、流量管理上的优势，同时，它还兼顾了第三层路由、寻径灵活的优势。

MPLS之所以吸引了学术界、产品制造商和运营商的注意力，除去**MPLS**的快速转发将为运营商和大型企业带来众多的利益外，最重要的原因它承诺为**IP**网络提供类似于**ATM**网络的控制和安全性，能够开展虚拟专用网（**VPN**）业务，做**IP**网上的流量工程（**TE**）以及提供服务质量（**QoS**）保证等。也就是说，**MPLS**既可以克服**IP**的一些致命缺点，又可以利用**IP**的很多优点。

电信级的可靠性是运营商永恒的要求，新型的核心路由器引入了先进的冗余备份方案，例如交换矩阵“1+1”备份、接口板“1:N”备份，从而确保了网络无中断运行。在电信级**IP**网络实际运行中，设备的调整是不可避免的，核心路由器的所有接口板都具有热插拔功能。由于有冗余备份方案，又采用交换式背板，因此热插拔不会造成数据包丢失。但是如果采用高速缓存的方式，就无法保证数据安全转发，给维护工作带来麻烦。例如，**A**接口板的包可能因为网络拥塞放在**B**接口板上备份，一旦对**B**板进行热插拔，数据包就会出现丢失。操作系统是高端路由器的软件核心，主要实现路由引擎功能，所以操作系统的稳健性直接关系到系统的性能。这要求软件不仅支持的路由协议最为全

面，更为重要的是，当网络出现故障时，软件系统应有很强的路由“学习”能力，可以在很短的时间内建立最优的路由表，便于最大限度地缩短故障时间，恢复运营服务。

MPLS将IP的灵活性和帧中继、ATM等面向连接网络的QoS保证特性有效地结合在了一起，这对于IP的进一步广泛应用无疑有着巨大的推动作用。

4.多播技术

多播（Multicast）主要用于视频会议等应用场合，这种应用需要同一份数据同时发送给多个用户。多播包的目的地地址使用D类IP地址，即224.0.0.0~239.255.255.255这个地址段。每个多播地址代表一个多播组，而不是一台主机。IGMP（Internet组管理协议）用于控制用户加入或离开多播组，多播路由协议则用于建立多播路由表（或称多播树）。

如果一个局域网中有一个用户通过IGMP宣布加入某多播组，则局域网中的多播路由器就将该信息通过多播路由协议进行传播，最终将该局域网作为一个分枝加入多播树。当局域网中的所有用户退出该多播组后相关的分枝就从多播树中删掉。多播网中可能有不支持多播的路由器，此时多播路由器使用IP over IP的隧道方式将多播包封装在单

播IP包中透传给相邻的多播路由器。相邻的多播路由器再将单播IP头剥掉，然后继续进行多播传输。

5.网管系统

网管系统在网络运营中起着非常重要的作用。方便、强大的网管系统可以协助用户有效地管理网络和降低网络维护费用。网管协议非常多，与路由器产品相关的网管协议主要有**SNMP**（简单网络管理协议）、**RMON**（远程镜像）等，其中**SNMP**最常见。**SNMP**采用代理（**Agent**）工作方式，设备侧（路由器上）运行**Agent**，网管站运行管理软件。代理的作用包括收集路由器统计数据（如端口收发报文总数等）和状态信息（如端口地址等），回答网管站对这些信息的查询；传达网管站的设置命令，如**TCP**连接复位、配置端口**IP**地址、发生异常事件时主动向网管站报告等。

7.7.3 三层交换机

早期的网络中一般使用二层交换机来搭建局域网，而不同局域网之间的网络互通由路由器来完成，如图7-50所示。那时的网络流量中，局域网内部的流量占了绝大部分，而网络间的通信访问量比较少，使用少量路由器已经足够应付了。

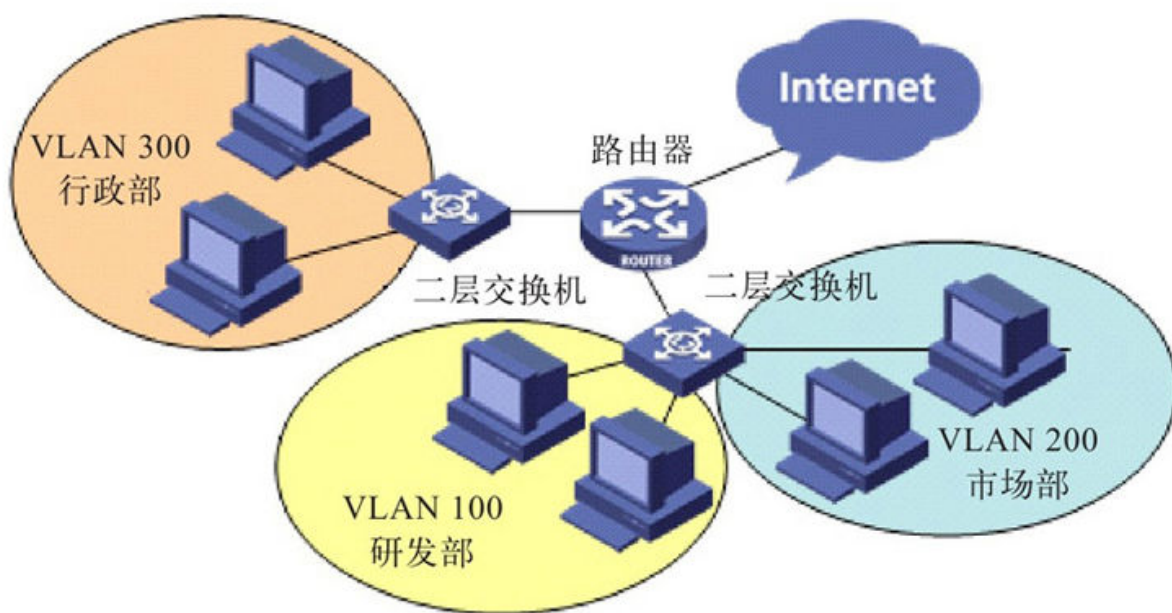


图 7-50 二层交换机+路由器组网基本结构

随着数据通信网络范围的不断扩大，网络业务的不断丰富，网络间互访的需求越来越大，而路由器由于自身成本高、转发性能低、端口数量少等特点无法很好地满足网络发展的需求。我们知道，路由器主要是通过IP转发（三层转发）来实现不同网络间的互连，那么是否

能够将交换机的高性能应用到三层转发中去呢？答案是肯定的，三层交换机就是这样一种实现了高速三层转发的设备。其典型网络结构如图7-51所示。

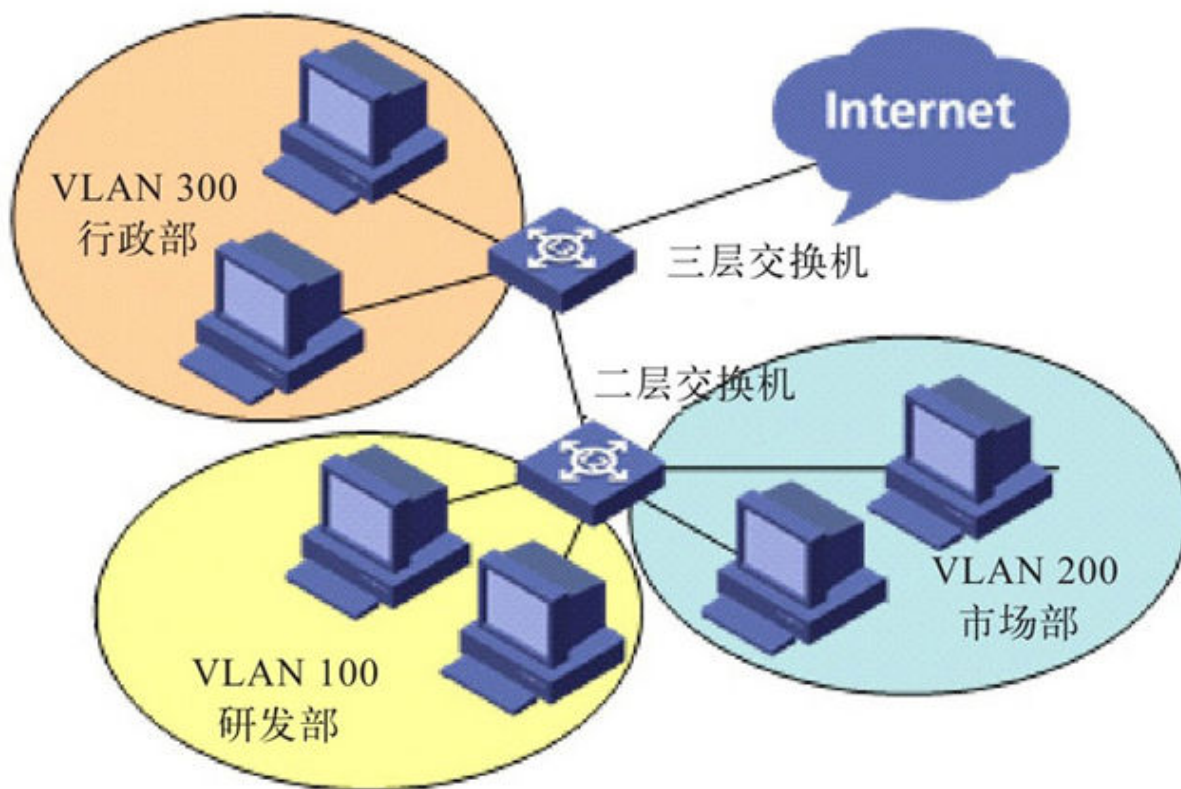


图 7-51 三层交换机组网基本结构

大多数三层交换机采用ASIC硬件芯片来完成转发，ASIC芯片内部集成了IP三层转发的功能，包括检查IP报文头、修改存活时间（TTL）参数、重新计算IP头校验和、封装IP包的数据链路等。

7.7.4 三层交换机硬件结构

总体来说，目前的三层交换机在三层路由模块中有三种类型，即纯硬件模块、纯软件模块和软/硬结合模块三大类。目前最常见的还是软/硬件结合的三层路由模块。

1. 纯软件路由模块的三层交换机

在早期，以及目前比较低档的三层交换机中基本上都是采用纯软件路由模块来实现三层交换功能的。这类三层交换机的基本内部结构如图7-52所示。在这种三层交换机中，是通过CPU调用相关的软件功能，查询存储在内存中的CAM（Content Addressable Memory，内容可寻址记忆）表来获取对应目的IP地址和MAC地址。

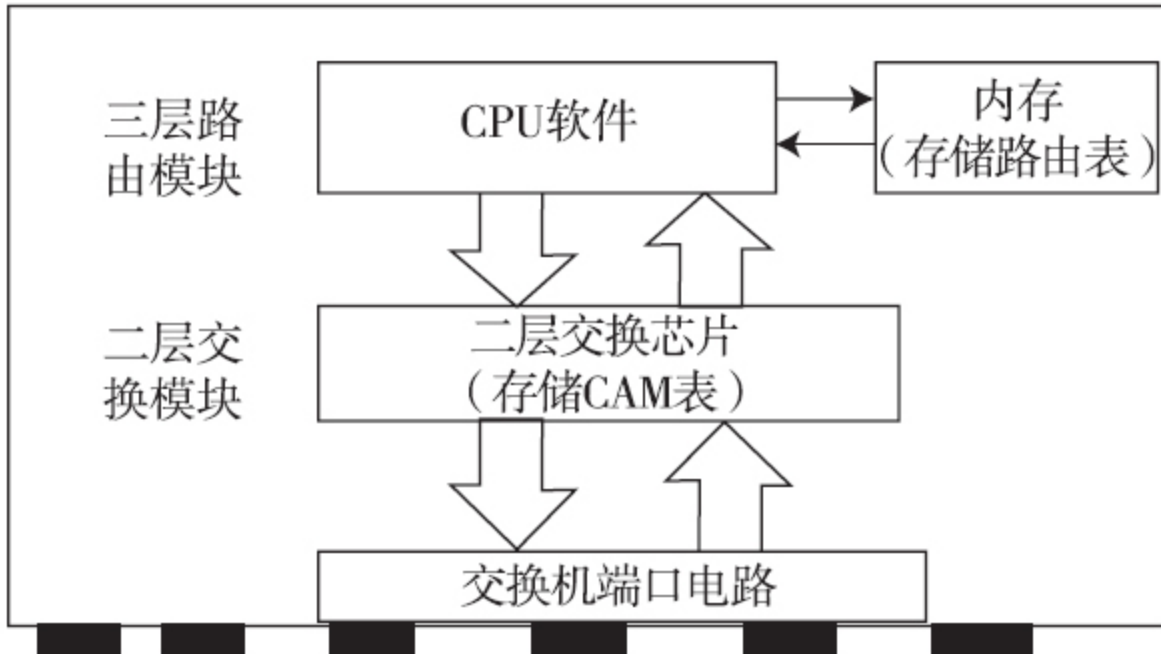


图 7-52 纯软件路由模块的三层交换机基本内部结构

2. 纯硬件路由模块的三层交换机

纯硬件的三层交换技术相对来说技术复杂、成本高，但是速度快、性能好、带负载能力强。其原理是采用专门的ASIC芯片进行路由表的查找和刷新，如图7-53所示。

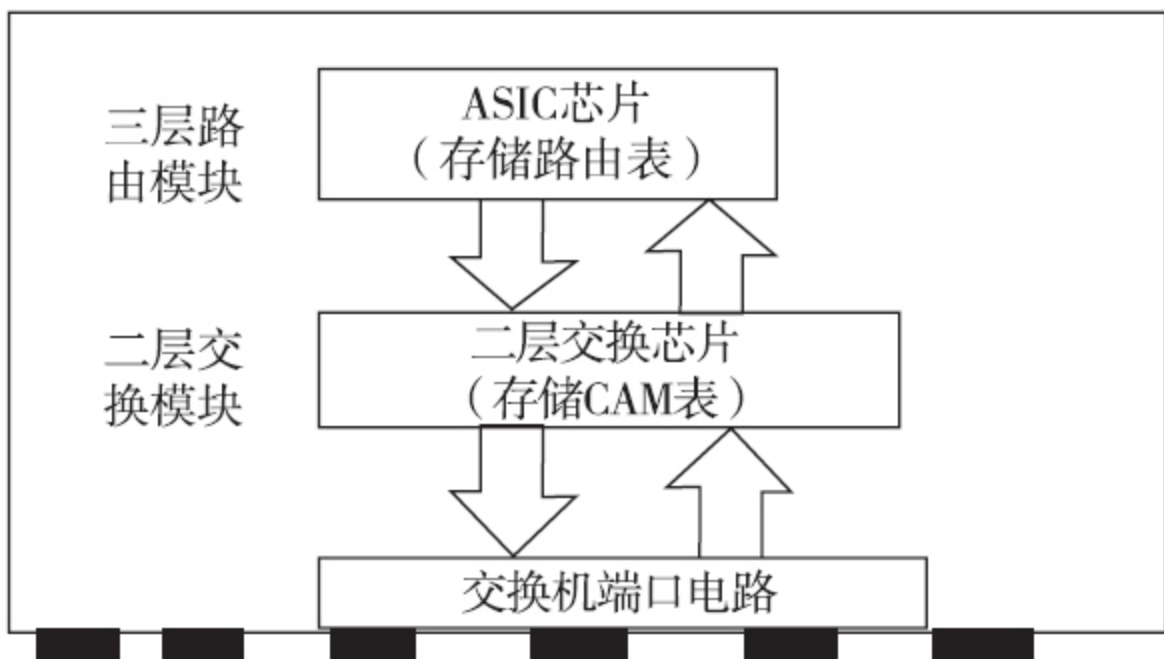


图 7-53 纯硬件路由模块的三层交换机基本内部结构

这种纯硬件路由模块的三交换机在数据交换原理方面与纯软件路由模块的三层交换机基本一样，不同的只是这里在进行路由转发时采用的是ASIC芯片，路由表信息也是存储在ASIC芯片中，执行效率更高。

3.软/硬件结合路由模块的三层交换机

目前纯软件或者纯硬件路由模块的三层交换机都比较少见，比较常见的是两者的结合的三层交换机，在ASIC芯片中同时存储二层CAM表和三层转发表，在内存中存储软件路由表和ARP表，由CPU调用，既利用硬件模块的高速交换性能，又利用软件模块的灵活性。

以上ASIC芯片用来完成主要的二、三层转发功能，内部包含用于二层转发的MAC地址表以及用于IP转发的三层转发表；CPU主要用于转发的控制，维护一些软件表项（包括软件路由表、软件ARP表等），并根据软件表项的转发信息来配置ASIC的硬件三层转发表。当然，CPU本身也可以完成软件三层转发，这就是前面介绍的纯软件三层路由模块的三层交换机。

从三层交换机的结构和各部分作用可以看出，真正决定高速交换转发的是ASIC中的二、三层硬件表项，而ASIC的硬件表项来源于CPU维护的软件表项。

7.7.5 三层交换原理

二层交换机的二层数据交换一般都是使用ASIC（Application Specific Integrated Circuit，专用集成电路）的硬件芯片中的CAM表来实现的，因为是硬件转发，所以转发性能非常高。而三层交换机的三层转发也是依靠ASIC芯片完成的（路由器的路由功能主要依靠CPU软件进行的），但其中除了二层交换用的CAM表外，还保存有专门用于三层转发的三层硬件转发表。

三层交换机的三层交换原理比较复杂，不同网络环境下不同厂家的三层交换机的三层交换流程都不完全相同。图7-54所示的仅是一个直接连接在一台三层交换机上的两个不同网段主机三层交换的基本流程，其各主要步骤解释如下：

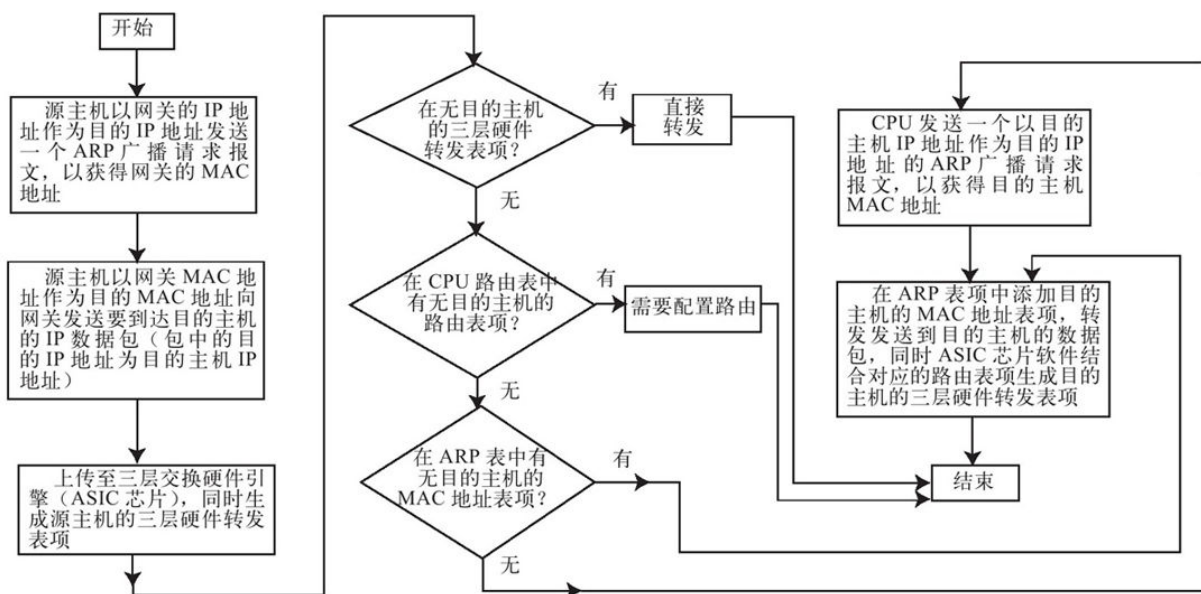


图 7-54 三层交换基本流程

1) 源主机在发起通信之前, 将自己的IP地址与目的主机的IP地址进行比较, 如果源主机判断目的主机与自己位于不同网段时, 则它需要通过网关来递交报文, 所以它首先需要通过一个ARP请求报文获取网关的MAC地址 (在源主机不知道网关MAC地址的情形下), 即源主机先发送ARP请求帧以获取网关IP地址对应的MAC地址。

2) 网关在收到源主机发来的ARP请求报文后以一个ARP应答报文进行回应, 在应答报文中的源MAC地址就是网关的MAC地址。

3) 在得到网关的ARP应答后, 源主机再用网关MAC地址作为报文的源MAC地址, 以源主机的IP地址作为报文的源IP地址, 以目的主机的IP地址作为目的IP地址, 先把发送给目的主机的数据发给网关。

4) 网关在收到源主机发送给目的主机的数据后, 由于查看得知源主机和目的主机的IP地址不在同一网段, 于是把数据报上传到三层交换引擎 (ASIC芯片), 在里面查看有无目的主机的三层转发表。

5) 如果在三层硬件转发表中没有找到目的主机的对应表项, 则向CPU请求查看软件路由表, 如果有目的主机所在网段的路由表项, 则还需要得到目的主机的MAC地址, 因为数据包在链路层是要经过帧封装的。于是三层交换机CPU向目的主机所在网段发送一个ARP广播请求包, 以获得目的主机MAC地址。

6) 交换机获得目的主机**MAC**地址后，向**ARP**表中添加对应的表项，并转发由源主机到达目的主机的数据包。同时三层交换机三层引擎会结合路由表生成目的主机的三层硬件转发表。

以后到达目的主机的数据包就可以直接利用三层硬件转发表中的转发表项进行数据交换，不用再查看**CPU**中的路由表了。

以上流程适用位于不同**VLAN**（网段）中的主机互访的情况，这时用于互连的交换机做三层交换转发。

7.7.6 三层交换示例

在三层交换中，同一交换机上的不同网段主机通信和不同交换机上的不同网段主机通信的基本原理是一样的，只是具体流程有所区别。本节仅以比较简单的“同一交换机上的不同网段主机通信”这种情形来解释上节介绍的三层交换原理。

如图7-55所示，通信的源、目的主机连接在同一台三层交换机上，但它们位于不同VLAN（即位于不同网段）。对于三层交换机来说，这两台主机都位于它的直连网段内，它们的IP对应的路由都是直连路由。图中已标明了两台主机的MAC地址、IP地址、网关IP地址（也就是对应VLAN接口IP地址），以及三层交换机的MAC地址。

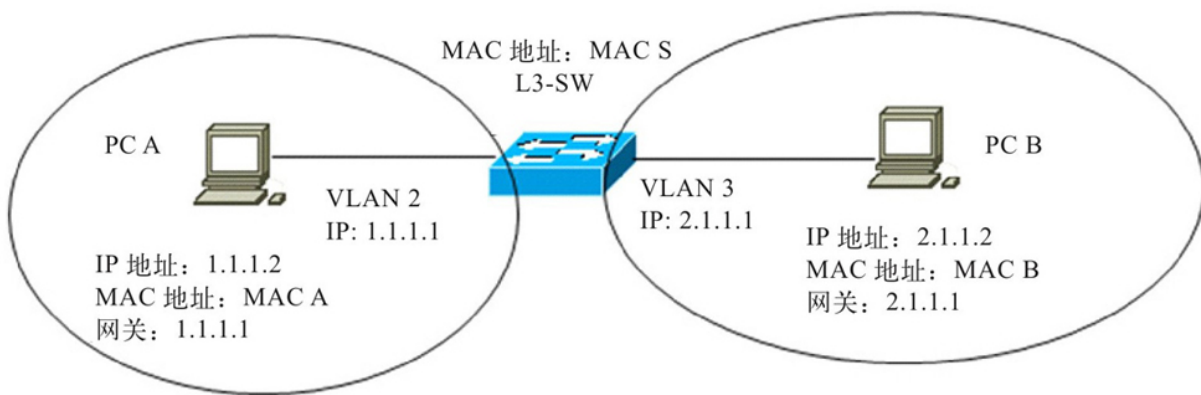


图 7-55 同一交换机上不同网段主机间的三层交换示例

说明 本示例中虽然划分了VLAN，但是在下面的数据转发流程中我们并没有提到VLAN标记，那是因为在在本示例中，通信双方主机都连

接到同一个三层交换机上，端口类型均为Access类型，发送和接收的数据帧都是无VLAN标记的。在下节介绍的示例中，在数据的转发过程中将会涉及VLAN标记问题。

当PC A第一次向PC B发向数据包时，数据包的传输流程如下（假设三层交换机上当前还未建立任何硬件转发表项）：

1) PC A首先检查出目的IP地址2.1.1.2（PC B）与自己不在同一网段，觉得就这样直接发送肯定不行，于是把要发送的数据包先缓存起来。它知道这个数据包必须经过网关来转发，所以先查看一下自己是否已经知道网关的MAC地址（也就是在PC A主机的ARP表中查看是否有与网关IP地址对应的MAC地址表项）。如果有，则直接把要发送给PC B的数据包在封装成数据帧时，把“目的MAC地址”字段的值设为网关的MAC地址（也就是三层交换机的MAC地址MAC S，交换机上各端口、各VLAN是共享一个或多个MAC地址的）。

2) 如果PC A在自己的ARP表中没有找到网关MAC地址，则先向网关发出（其实是会向本VLAN内所有节点发出）一个ARP广播请求报文，以获取网关IP地址1.1.1.1所对应的MAC地址。此ARP请求报文的源MAC地址字段是PC A的MAC地址MAC A，目的MAC地址因为未知，故以全0格式填充；源IP地址和目的IP地址字段分别填上PC A的IP地址（1.1.1.2）和网关的IP地址（1.1.1.1）。

ARP请求报文向下传输到了以太网数据链路层后被再次封装成以太网帧，以太网帧头中的源MAC地址字段值仍为PC A的MAC地址MAC A，目的MAC地址字段值为广播MAC地址FF-FF-FF-FF-FF-FF，帧类型字段填上ARP的协议号0x0806。有关ARP报文和ARP帧格式参见7.3节。

3) 三层交换机在收到PC A发来的ARP请求报文后，检查请求报文发现被请求IP地址（也就是“目的IP地址”）是自己的三层接口IP地址，于是向PC A发回一个ARP应答报文，并将对应的三层接口MAC（MAC S）填充在应答报文中的目的MAC地址字段其中。同时通过对PC A发送的ARP请求报文分析，把PC A的IP地址与MAC地址对应关系（1.1.1.2 <==> MAC A）记录到自己的ARP表中去，然后把PC A的IP地址（作为目的IP地址）、MAC地址（作为下一跳MAC地址），以及与交换机直接相连的端口号等信息下发到三层交换机ASIC芯片中的三层硬件转发表。此时在三层硬件转发表中就有了第一个转发表项，即PC A的转发表项。

说明 在三层交换机中，最关键的就是它有一个专门用于三层转发的三层硬件转发表，它和ARP表之间有联系，但也有区别。ARP表中只是IP地址和MAC地址的映射关系，不包括转发出口，也不包括对应的VLAN ID，而三层硬件转发表中则包括了全部这些，形成一个目的IP地址、VLAN ID、端口和下一跳MAC地址的关系表项。因为在进

行三层转发时，改变的是帧封装后的源和目的MAC地址这两个字段，原来输入IP包中的目的MAC地址作为转发的下一跳MAC地址，原来的源MAC地址改为三层交换机自身的MAC地址，源和目的IP地址都不变（因为这是封装在帧的数据部分）。另外，三层转发表是存储在ASIC硬件芯片上的，直接由ASIC芯片调用，而ARP表存储在内存中，由CPU软件调用。但三层硬件转发表项还是由CPU提供的。

4) PC A在收到网关的ARP应答报文后，把要发送给PC B的数据包中经过帧封装后的目的MAC地址修改为网关MAC地址（MAC S1），其它不变，先把数据包发给网关（三层交换机）。

5) 三层交换机在收到这个数据包后，因为目的MAC地址为交换机自己的MAC地址，而且目的IP地址和源IP地址不在同一网段，所以会直接提交到负责三层交换的ASIC芯片，根据包中的目的IP地址（PC B的IP地址2.1.1.2）在三层硬件转发表中查看有无对应表项，因为是第一次通信，所以结果是查找失败，于是将数据包再转送到CPU中进行软件路由处理。

6) CPU同样会根据包中的目的IP地址查找其软件路由表，发现匹配了一个直连网段（PC B对应的网段），于是继续在ARP表中查找对应的MAC地址项。同样是由于是第一次查找，所以仍然查找失败。如果在ARP表中找到了对应的MAC地址，则数据可以直接由软件路由表转发了。

7) 下面仍以在ARP表中也没找到PC B对应的MAC地址为例进行介绍。此时三层交换机CPU会在PC B所在网段的AN 3中所有端口发送一个ARP广播请求报文，查找目的IP地址为2.1.1.2所对应MAC地址。报文经过帧封装后的源MAC地址是三层交换机的MAC地址（MAC S），目的MAC地址全为0，源IP地址是VLAN 3网段的网关IP地址（2.1.1.1），目的IP地址是PC B的IP地址（2.1.1.2）。

8) PC B在收到三层交换机CPU发送的ARP请求报文后，检查发现被请求的IP地址是自己的IP地址，于是发送一个ARP应答报文，并将自己的MAC地址（MAC B）包含在其中。同时，将三层交换机上VLAN 3网段的网关IP地址与MAC地址的对应关系（2.1.1.1 <==> MAC S）记录到自己的ARP表中。

9) 三层交换机CPU在收到PC B的ARP应答报文后，将其IP地址和MAC地址对应关系（2.1.1.2 <==> MAC B）记录到自己的ARP表中，把PC B的IP地址、MAC地址、进入交换机的端口号等信息下发到三层交换机的三层转发中。此时转发表中就有到达PC A和PC B这两条对应的表项了。

10) 三层交换机CPU根据获得的PC B的MAC地址和端口信息，以及软件路由表信息，把由PC A发来的IP数据包转发给PC B，这样就完成了PC A到PC B的第一次单向通信。

由于芯片内部的三层引擎中已经保存了从PC A到达PC B的完整转发路径信息，所以以后PC A与PC B之间进行通信，或其它网段的站点想要与PC A或PC B进行通信时，三层交换机的ASIC芯片就会直接把包从对应的三层硬件转发表项中指定的端口转发出去，而不必再把包交给CPU进行路由处理。这就是所谓的“一次路由（指通过CPU路由表查到了对应的直连网段），多次交换”的原理，这大大提高了转发速度。

7.7.7 三层交换机和路由器的主要区别

三层交换机和路由器都具有“路由”功能，看起来好像功能基本一样，其实两者无论从结构上，还是从应用环境，两者都存在明显区别的。具体区别主要体现在以下几个方面：

(1) 三层功能实现机制不同

有许多人不明白，它们都可以提供三层路由功能，既然有了路由器，为什么还要三层交换机，或者反过来问，既然有了三层交换机，为什么还要路由器？之所以有这样的疑问，其实是不明白三层交换机和路由器的三层功能实现机制的原因造成的。

路由器中的三层功能纯粹依靠所安装的网络操作系统（如Cisco路由器中的IOS系统，华为路由器中的VRP系统，H3C路由器中的Comware系统），CPU通过调用集成在这些操作系统的路由协议中的路由表来实现路由功能。所有经过路由器的数据包都必须通过路由功能来实现转发。

而三层交换机的三层功能包括两个方面：一是与路由器一样，它的CPU也可以调用所安装的网络操作系统中的软件路由器功能，二是它像二层交换机一样也有专门用于数据交换的硬件——ASIC芯片，在

它里面同时提供了二层交换模块（含有**CAM**表）和三层交换引擎，其中的三层交换引擎中建立有专门用于三层交换的三层硬件转发表。在三层交换机中，不同网段的通信采用的是“一次路由，多次交换”机制（新型的交换机还有一些快速交换技术，如Cisco的CEF（Cisco Express Forwarding,Cisco特快交换）），也就是到达某个目的主机的数据包只第一次采用路由方式，在建立了三层转发表后就会直接通过三层转发表进行二层转发了，大大提高了转发的效率。

三层交换机和路由器三层功能实现机制的不同也决定了它们在其他许多方面的不同。

（2）适用环境不同

三层交换机是由二层交换机发展而来的，就是想在保留二层交换机的二层交换功能的基础上再提供三层路由功能。这样一来，三层交换机可以有許多用于连接主机及各类其他设备的端口，在局域网内为各节点提供快速的二层数据交换。因为三层交换机在其ASIC芯片中也有像二层交换机一样的硬件**CAM**表。同时它又可以为局域网中不同**VLAN**，或者不同子网提供三层路由和三层转功能，也就是兼具二层交换机和路由器的双重功能，应用更加灵活。

或许你马上就又会问，用路由器来连接局域网中的不同**VLAN**或子网不就行了？这是可以的，但是路由器采用的是软件路由方式，路

由性能比较低下，主要适用于低速、互访并不频繁的WAN网络中；而在局域网中，不同VLAN或不同子网中互访非常频繁，低速的路由性能很难满足实际的用户需求，特别是在较大的局域网中。三层交换机的三层软件功能是基于硬件芯片的，三层转发性能更高，主要用于企业内部以太网中不同VLAN、不同子网间的三层数据交换（一般不提供WAN接入功能），由于现在其一般是采用硬件数据交换方式，故性能更高。

（3）多协议支持能力不同

尽管路由器在三层功能实现效率上比三层交换机差些，但并不是说三层交换机就可以全面取代路由器。因为三层交换机目前仅支持IP网络层协议，所以仅可适用于TCP/IP网络，所以才把三层交换机说成是IP交换机。而路由器在多协议支持方面要远好于三层交换机，因为路由器主要应用于WAN中，WAN中的网络类型可能非常复杂，所以需要支持更多的网络层协议。

（4）路由能力不同

虽然说三层交换机在三层功能实现上除了路由功能外，还有效率更高的硬件三层转发功能，但仅就路由功能来说，路由器的路由能力还是要远好过三层交换机的，因为它可以支持更加复杂的网络路由，可以支持更大的路由表项，更加复杂的路由表计算。另外，由于三层交

交换机的三层转发功能主要依靠ASIC芯片，它是硬件，不容易升级、更新，而路由器的路由功能则是领先软件的网络操作系统，更容易升级、更新。

综上所述，就目前来说三层交换机和路由器可以说是各有优势，但它们的主要应用领域各不相同，三层交换机主要用于企业IP局域网中不同VLAN或不同子网间的三层通信，而路由器则主要用于WAN中的网络互连。谁也不能完全替代谁。

第8章 IP地址和子网

我们已对IPv4和IPv6协议有了一个比较全面的了解，前面着重介绍了它们各自的报文格式、IPv4报文封装和解封装，IPv4报文分段与重组，以及IPv4和IPv6协议簇中主要的子协议用途、报文格式等。

本章要专门介绍在IP网络中我们每天都要配置的IPv4和IPv6地址，其中主要包括IPv4地址和IPv6地址的表示形式和分类，各类IPv4、IPv6地址结构，以及在进行网络规划和设计中经常需要考虑的IPv4子网划分与聚合方法，IPv6地址的自动配置过程。这些都是非常基础却非常重要的知识和技能，不要说网络职业人士，就是其他专业的IT职业人员都必须全面了解和掌握，因为无论是在一般计算机配置，还是在网络规划、设计与组建过程中，IP地址的配置与管理都是必不可少的。

8.1 IPv4地址

OSI/RM的网络层和TCP/IP协议体系结构的网际互连层最重要的一个协议就是IP协议，目前正处于IPv4和IPv6这两个版本交接、过渡时期，所以本章会同时对这两种版本的IP地址及相关知识进行系统介绍。本节先介绍目前仍为主流的IPv4地址。

8.1.1 IPv4地址基本格式

IPv4使用32位（4字节）地址，因此整个地址空间中有4 294 967 296（ 2^{32} ）个地址，也就是近43亿个地址。不过，其中一些地址是为特殊用途保留的，如局域网专用地址（约1800万个地址）和组播地址（约2700万个地址），这样一来可直接在广域网上使用的、路由的公网IP地址数量就更加少了。

说明 公网IP地址是指可以在广域网上直接使用，直接被路由（也就是可以被指路径查找到），并需向IP地址管理机构申请、注册、购买，且全球唯一（不存在多个用户拥有、使用相同的公网IP地址的情况）的IPv4地址。打个比方来说，公网IP地址就相当于我们公民的身份证号，每个身份证号都是全国唯一，并且通过这个身份证号，就可以查到我们的基本信息，找到我们。公网IP地址直接分配给互联网上的主机、服务器或其他设备，可以通过它在全球范围内找到对应的主机、服务器和设备。如各大企业网站通常都是直接使用公网IP地址的。

与公网IP地址相对应的自然是私网IP地址了，又称专用网络IP地址或者局域网IP地址。私网IP地址是指仅可以在各用户自己的局域网内部使用，且不同用户可以重复使用，无须向IP地址管理机构申请、注册，也无须购买的IPv4地址。私网IP地址就相当于我们企业内部的

员工编号，仅在企业内部有用，不能通过这个员工编号来在全国范围找到我们。企业内部局域网使用的就是私网IP地址，具体有哪些IP地址属于私网IP地址将在本章后面介绍。

随着公网IP地址不断被分配给最终用户，IPv4地址枯竭问题也在随之产生。虽然基于可变长子网掩码（VLSM）、无类别域间路由（CIDR）和网络地址转换（NAT）的地址结构重构显著地减少了地址枯竭的速度，但在2011年2月3日，在最后5个地址块被分配给5个区域互联网注册管理机构之后，IANA的主要地址池空了，所以现在正在积极推行IPv6，具体将在本章后面介绍。

IPv4地址在计算机内部是以二进制形式表示的，每个地址都有32位，由数字0和1构成。在这32位的二进制数中，其实每个8位之间并没有我们所看到的那个用来分隔各段的一个小圆点，只是为了方便我们自己阅读，在每个字节间用一个小圆点分隔。因为整个IP地址有32位，无论是书写还是记忆都不方便，于是我们在日常的IP地址管理中把这个32位长的二进制IP地址分段转换成对应的十进制，在每个字节间用小圆点分隔。引用某个IPv4地址时，可使用W.X.Y.Z的点分十进制表示形式，如192.168.1.10等。图8-1所示为IPv4地址的基本结构，图中的W、X、Y、Z每个都代表一个8位二进制组。

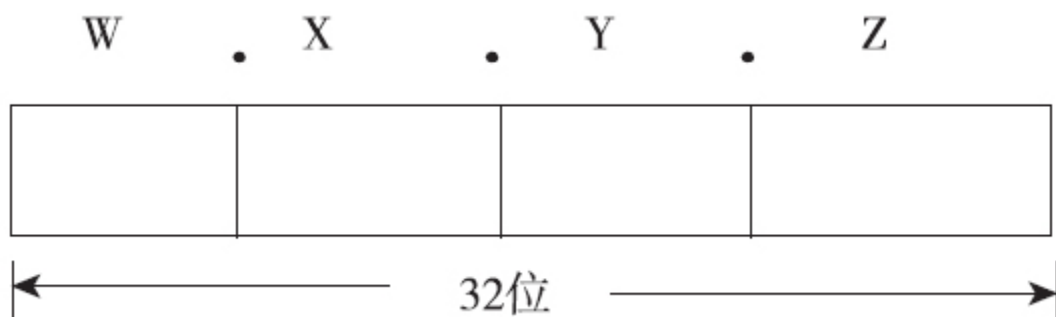


图 8-1 点分十进制表示形式的IPv4地址

由前面介绍的数制转换内容可以知道，每个8位二进制所能表示的最大数就是 $2^8 - 1 = 256 - 1 = 255$ （最小数是0），所以IPv4地址转换成十进制数后，每段8位二进制组的取值范围就是0~255。因为IP地址在计算机中是以二进制表示的，32位就相当于4字节，所以在IPv4协议数据报格式（参见7.3.4节）中，无论是源IP地址字段，还是目的IP地址字段都占4字节的。

当然，IPv4地址还可表示成八进制、十六进制，只是我们平时不用而已，具体的转换方法参见本书第1章相关内容。

8.1.2 子网掩码

我们在为设备配置IP地址时，通常是不能仅配置IPv4地址，而必须同时配置所谓的子网掩码，如图8-2所示。那么子网掩码是什么？它又有什么作用？

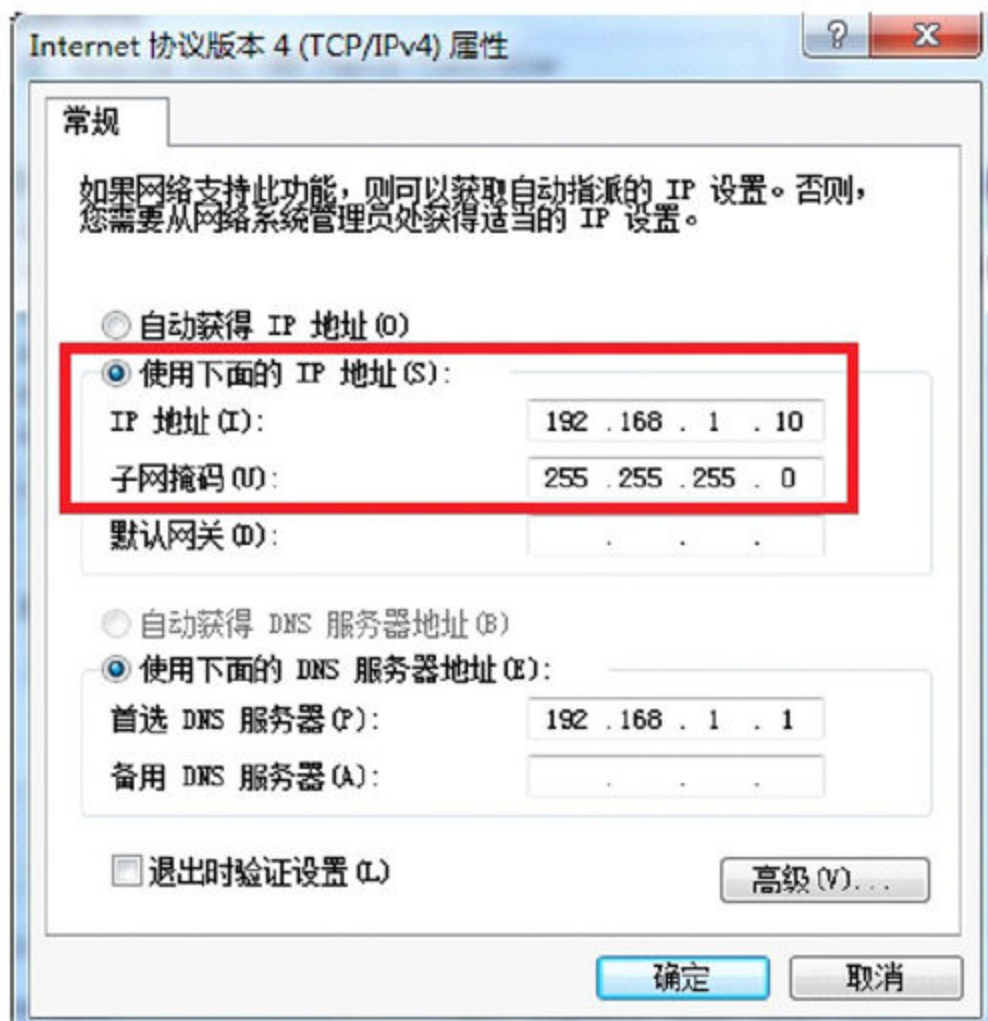


图 8-2 同时的IPv4地址和子网掩码

要想理解什么是子网掩码，就不能不先了解IPv4地址的构成。互联网是由许多小型网络构成的，每个网络上都有许多主机，这样便构成了一个有层次的结构。IPv4地址在设计时就考虑到地址分配的层次特点，将每个IP地址都分割成网络ID和主机ID两部分，以便于IPv4地址的寻址操作。那么IPv4地址的网络ID和主机ID各是多少位呢？如果不指定，在寻址时就不知道对应IPv4地址中哪些位代表网络ID、哪些代表主机ID，这就需要通过这里所说的子网掩码来实现了。

与二进制IPv4地址相同，子网掩码也由1和0组成，且长度也是32位，我们也可以把它分成网络ID和主机ID两部分，且各自的长度与IPv4地址的网络ID和主机ID部分对应相等。但子网掩码中的网络ID部分全是1，1的数目等于网络ID的长度；主机ID部分全是0表示，0的数目等于主机ID的长度。如图8-3所示的是一个网络ID长度为16的子网掩码示例。这样做的目的是为了在寻址过程中使子网掩码与对应的IPv4地址做逻辑与运算时用0遮住IPv4地址中原主机ID部分（因为0与任何数相与结果都是0），而不改变原网络ID部分（因为1与任何数相与的结果都不改变原来的值），这样就一来就可以很容易确定对应目的IPv4地址所在的网络了，确定了网络，也就确定了主机，因为在IPv4地址中除了网络ID部分就是主机ID部分。

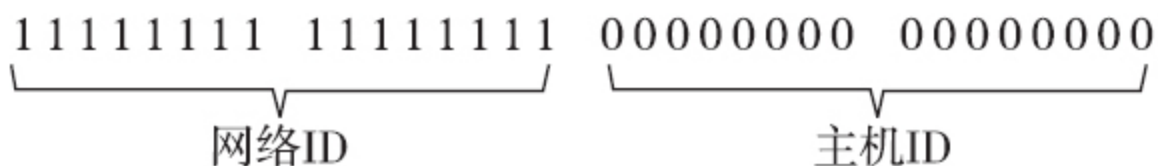


图 8-3 子网掩码示例

子网掩码不是一个地址，但是可以确定一个IPv4地址中的哪一部分是网络ID，哪一部分是主机ID，连续1的部分就代表网络ID，连续0的部分代表主机ID。子网掩码的作用就是获取主机IPv4地址中的网络地址信息，用于区别主机通信不同情况，选择不同路由。子网掩码一旦设置，对应IPv4地址中的网络ID和主机ID部分就固定了。

与IPv4地址一样，子网掩码也可以转换成点分十进制形式。根据子网掩码格式可以发现，子网掩码有0.0.0.0；255.0.0.0；255.255.0.0；255.255.255.0；255.255.255.255五种，其中0.0.0.0掩码代表任意网络的掩码，如我们在设置默认路由时，不仅IP地址为0.0.0.0，子网掩码也为0.0.0.0；A类地址的默认子网掩码为255.0.0.0；B类地址的默认子网掩码为255.255.0.0；C类地址的默认子网掩码为255.255.255.0；而255.255.255.255可以看作是单一主机网络，代表这个网络就这一个IPv4地址，在配置ACL（访问控制列表）时，如果控制的是一台主机，则对应的子网掩码也为255.255.255.255。有关A、B、C地址的分类将在下节介绍。

8.1.3 IPv4地址的基本分类

IPv4地址共有 2^{32} 个，最初把一个IPv4地址分成两部分：“网络识别码”在地址的最高一个字节中，“主机识别码”在剩下的部分中。这样划分的话，就使得最多只能分配给256个网络，显然这样是远远不够的。

为了克服这个限制，在随后出现的分类网络中，地址的高位字节被重定义为网络的类别（即网络ID），共五个：A、B、C、D和E。A、B和C类用于单播通信中设备IP地址分配；D类属于组播地址，用于组播通信；E类是保留地址。它们均有不同的网络类别（也就是网络ID）长度，剩余部分用来识别网络内的主机（称为主机ID）。网络ID用来确定每类网络中有的网络数，而主机ID用来确定每个网络中的IP地址数。下面分别介绍这五类地址的结构。

1.A类IPv4地址

A类IPv4地址结构如图8-4所示，其中网络ID占用最高一个字节，也就是第一个二进制8位组，而主机ID则占用剩余三个字节，也就是后面的三个二进制8位组。

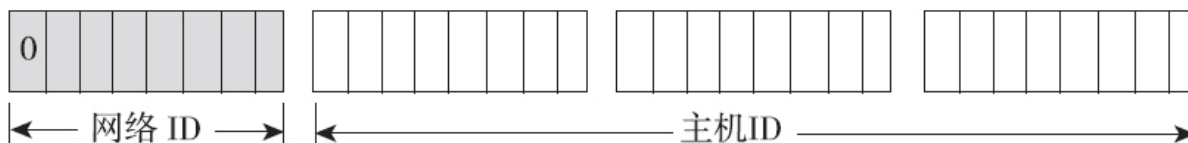


图 8-4 A类地址的结构

在分类中规定，A类IPv4地址中网络ID的最高位固定为0，后面的7位可变。这样一来，A类网络的总数从256（ 2^8 ）个减少到128（ 2^7 ）个。但实际上可用的只有126个，即整个IPv4地址中可构建126个A类网络，因为网络ID为0和127的A类网络不可用的。网络ID全为0的地址为保留地址，不能被分配；而网络ID为01111111（相当于十进制的127）的地址专用本地环路测试（也就是通常所说的环路地址），也是不能分配的。也就是凡是以0或者127开头的地址是不能分配给节点使用的。

又因为A类IPv4地址中主机ID有24位，所以可用的主机ID数，也就是可以每个A类网络中拥有的IPv4地址数为166 777 216（ 2^{24} ）。但主机ID全为0的地址为网络地址，而主机ID全为1的地址为广播地址，不能分配给主机使用，所以实际上可用的地址数为166 777 214（ $166\,777\,216-2$ ）。A类网络中可以构建的网络数最少，但每个网络中拥有的地址数是最多的，也就是可以构建的网络规模最大，适用于大型企业和运营商。

A类IPv4地址的子网掩码为固定的255.0.0.0，因为子网掩码就是网络ID部分全为1，主机ID部分全为0，而A类地址中网络ID部分就是最高的那个字节。

2.B类IPv4地址

B类IPv4地址结构如图8-5所示，其网络ID占用最高的前两个字节，也就是第一个和第二个二进制8位组，而主机ID则占用剩余的两字节，也就是后面两个二进制8位组。

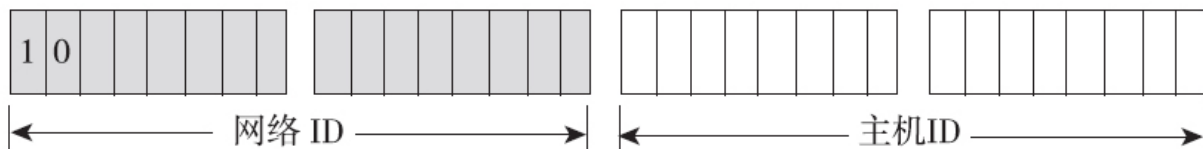


图 8-5 B类地址的结构

B类IPv4地址的网络ID的最高两位固定分别为1、0，后面的14位可变。由此可知B类网络的总数从65 536 (2^{16} ，也可写成 256×256)减少到16 384 (64×256)个；B类IPv4地址中主机ID为16位，所以可用的主机ID数，也就是每个B类网络拥有的IPv4地址数为65 536 (2^{16})个。同样因为主机ID全为0的地址为网络地址，而主机ID全为1的地址为广播地址，不能分配给主机使用，所以实际上可用的地址数为65 534 ($65\,536 - 2$)。

B类IPv4地址的子网掩码为固定的255.255.0.0，因为B类地址中网络ID部分是最高的两个字节，每个字节均为8个连续的1，转换成十进制后每个字节就是255了。

3.C类IPv4地址

C类IPv4地址结构如图8-6所示，其网络ID占用最高的前三个字节，也就是第一个、第二个和第三个二进制8位组，而主机ID只占用了最后的一个字节，也就是只有最后一个二进制8位组。

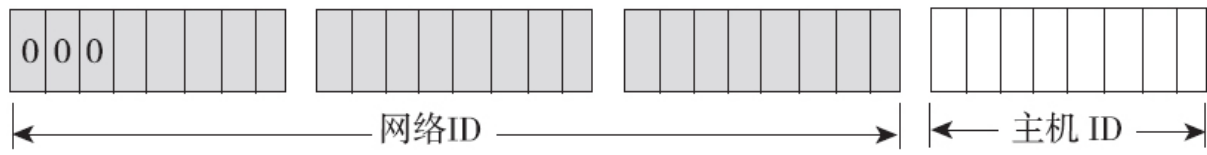


图 8-6 C类地址的结构

C类IPv4地址的网络ID的最高两位固定分别为1、1、0，后面的21位可变。由此得知C类网络总数从166 777 216 (2^{24} ，也可写成 $256 \times 256 \times 256$) 个减少到2 097 152 ($32 \times 256 \times 256$) 个。C类IPv4地址中主机ID仅为8位，所以可用的主机ID数，也就是每个C类网络拥有的IPv4地址数为256 (2^8) 个。同样因为主机ID全为0的地址为网络地址，而主机ID全为1的地址为广播地址，不能分配给主机使用，所以实际上可用的地址数为254 ($256-2$)。

C类单播地址的子网掩码为固定的255.255.255.0，因为C类地址中网络ID部分是最高的前三个字节，每个字节均为8个连续的1，转换成十进制后每个字节就是255了。

表8-1总结了A、B和C三类IPv4地址的主要特征（如图8-1所示）。

表 8-1 A、B 和 C 三类 IPv4 单播地址的主要特征

类别	w 的值	网络 ID 部分	主机 ID 部分	网络 ID 数	每个网络的主机 ID 数
A	1-126	w	x.y.z	126	16 777 214
B	128-191	w.x	y.z	16 384	65 534
C	192-223	w.x.y	z	2 097 152	254

4.D类IPv4地址

D类IPv4地址是组播地址，用于IPv4组播通信中。通过组播IPv4地址，组播时源主机（组播源）只需发送一份数据，就可以使对应组播组（组播组使用D类IPv4地址标识）中的一个或多个主机收到这份数据的副本的通信方式，但只有组播组内的主机可以接收该数据。

IP组播技术有效地解决了单点发送多点接收的问题，实现了IP网络中点到多点的高效数据传送，能够大量节约网络带宽、降低网络负载。还可以利用网络的组播特性方便地提供一些新的增值业务，包括在线直播、网络电视、远程教育、远程医疗、网络电台、实时视频会议等互联网的信息服务领域。

D类IPv4的地址结构如图8-7所示，规定在最高字节中前4位分别固定为1、1、1、0，整个组播地址范围为224.0.0.0~239.255.255.255。



图 8-7 D类IPv4地址结构

整个组播IPv4地址根据不同的应用环境和用途又可划分为预留组播地址、公用组播地址、临时组播地址、本地管理组播地址四大类。

（1）预留组播地址

预留组播地址（又称永久组播地址）就是由IANA保留不分配给特定用户使用，仅为公用的组播路由协议分配使用的组播地址，地址范围为224.0.0.0~224.0.0.255。使用这些预留组播地址的组播协议包括IGMP（Internet组管理协议）、CGMP（Cisco组管理协议）、IGMP Snooping（IGMP侦听）和PIM（协议无关组播）等。使用这段组播地址的IP包不被路由器转发。

在这个地址组段中，224.0.0.0不分配；224.0.0.1分配给本地组播网络所有支持组播的主机；224.0.0.2分配给本地组播网络中的所有组播路由器；224.0.0.4分配给本地组播网络中的所有DVMRP路由器；224.0.0.5分配给本地组播网络中的所有OSPF路由器；224.0.0.6分配给本地组播网络中的所有OSPF指定路由器（DR）；224.0.0.9分配给本地组播网络中的所有RIPv2路由器；224.0.0.10分配给本地组播网络中的所有IGRP路由器；224.0.0.13分配给本地组播网络中的所有PIMv2路由器；224.0.0.22分配给本地组播网络中的所有IGMPv3路由器。

（2）公用组播地址

公用组播地址就是在全球范围内可以直接在互联网上使用的组播地址，就像前面介绍的公网单播IPv4地址一样。公用组播地址范围为224.0.1.0~224.0.1.255，也是由IANA为提出申请并付费的用户分配的。

(3) 临时组播地址

临时组播地址就是由企业用户在本企业局域网内部使用的组播地址，地址范围为224.0.2.0~238.255.255.255，仅在本地局域网内有限，就像前面介绍的局域网IPv4地址一样。

(4) 本地管理组播地址

本地管理组播地址也是保留使用的，专用于局域网内部组播测试，地址范围为239.0.0.0~239.255.255.255，仅在特定的本地网络范围内有效。

当网络层收到组播数据报文时，根据组播目的地址查找组播转发表，对报文进行转发。在私网中，组播是不需要在工作站上配置的，只需要在网络中的路由器或者支持组播协议的三层交换机上进行配置。私网工作站中被分配的组播地址都是224.0.0.1，就像环路地址127.0.0.1一样，无须另外配置。只要在路由器中启用了组播协议后就可以对应加入到组播组中。公网中，工作站组播地址选择224.0.1.0~238.255.255.255范围中的一个即可。

另外，要注意的是，在进行组播通信时，在数据链路层目的MAC地址字段封装的也是组播MAC地址。IANA把01:00:5E开头的以太网MAC块作为组播地址对应的二层组播MAC地址。组播MAC地址的范围是01:00:5E:00:00:00~01:00:5E:7F:FF:FF（前24位为MAC头，固定不变，第25位为0），并要求将IPv4组播地址的后28位（因为最高的4位是固定不变的）映射到48位的MAC地址空间中。

具体的映射方法是将组播IPv4地址中的低23位放入MAC地址的低23位，如图8-8所示。至于为什么要映射后面的23位，原因就在于根据IANA给出组播MAC地址段是前3字节（也就是24位）来标识单位或者厂商，只有后面24位来和IP地址映射；而给定的地址空间后3字节的最高位相同，都为0，那么给定的MAC地址段内只有23位了，所以最终只能丢弃28位IPv4地址中的前5位，剩下的23位和MAC的23位相映射。注意，这个映射无须手动进行，在路由器启动组播协议，站点加入到组播组后就会自动生成。

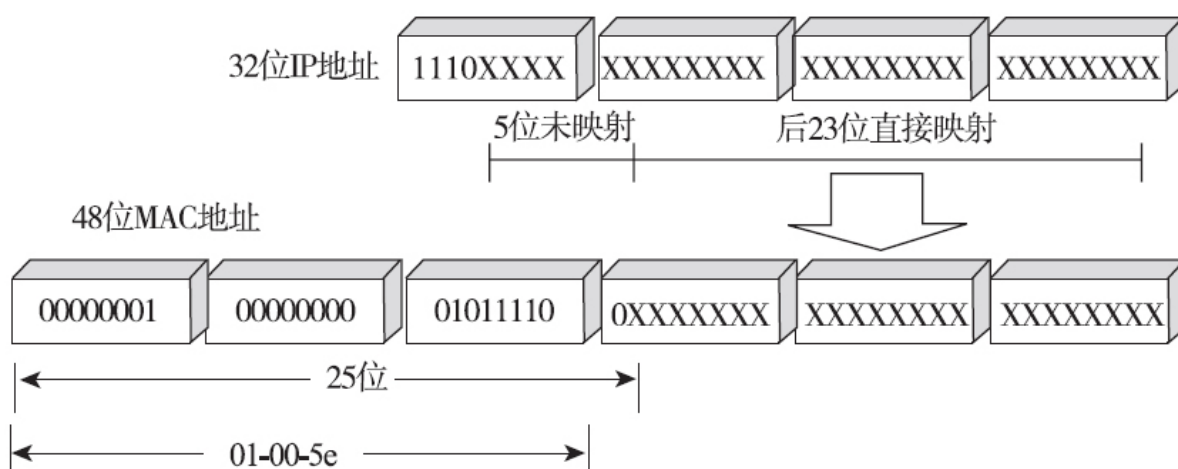


图 8-8 多播IP地址到MAC地址的映射

由于IPv4多播地址的后28位中只有23位被映射到MAC地址，这样会有32个（ 2^5 ，IPv4多播地址中有5位可变）IP多播地址映射到同一MAC地址上。

5.E类IPv4地址

E类IPv4属于IANA保留地址，不分配给用户使用，地址段范围为240.0.0.0~247.255.255.255，其特征是最高5位分别是1、1、1、1、0，如图8-9所示，也就是有27位是可变的。



图 8-9 E类IPv4地址结构

8.1.4 有类/无类IPv4网络

通过上节的介绍我们知道，尽管已针对整个IPv4地址空间进行分类，但每类网络的数量对于全球用户数量来说仍非常之少，特别是A类网络才区区的126个。之所以会出现这种现象，原因就在于每类网络的网络ID部分（也就是子网掩码部分）的位数是固定不变的，而且是字节的整数倍，如A类网络中的网络ID部分占1字节（8位），B类网络中的网络ID部分占2字节（16位）、C类网络中的网络ID部分占3字节（24位）。

为了解决这种可用网络数太少的问题，同时提高IP地址有效利用率，于是出现了一种称为VLSM（Variable Length Subnet Masking，可变长子网掩码）的技术，该技术可以使网络中IPv4地址的子网掩码所占位数不是固定的，如子网掩码位数可以不是以字节为单位的整数，如12、13、14、25、26、28、等均可。因为子网掩码位数不固定了，也就不能再依据子网掩码来进行分类了，所以采用VLSM后形成的网络称为无类网络，而以前传统的A、B、C类网络则相对应地被称为有类网络或标准网络。而我们在有类网络中通过VLSM技术改变子网掩码长度而划分的小网络称为子网。有关子网的划分将在本章后面具体介绍，在此仅介绍无类网络的一些基础知识。

这时你可能马上会问，**VLSM**能给我们带来什么好处？简单地说体现在三个方面：①用户可以根据网络规模大小选择拥有适当可用**IPv4**地址的**IPv4**地址段，这样可以减少**IPv4**地址的闲置和浪费，降低广播风暴产生的可能；②可以提高路由效率；③提高广域网中的网络安全。

举例来说，先说在局域网中。假设你公司的局域网节点数为**1000**个左右（也就需要**1000**个左右的**IPv4**地址），这时如果想把所有设备都放在一个网段中，这样各设备可以直接进行二层通信，无需三层网络设备。如果不采用**VLSM**，根据表**8-1**介绍的三类网络类型所对应拥有的**IPv4**地址数可以知道，你最佳的选择就是用一个**B**类地址网络，但它每个网络可以使用的**IPv4**地址总数达到了**65534**个，相对你只有**1000**个节点的网络来说，显然是大材小用了，必将造成有绝大部分地址在你的网络中根本用不上，白白空置了。这时假设你的公司又有新的局域网要组建，节点数在**300**个左右，根据表**8-1**可知，你同样得选一个**B**类地址段来进行**IP**地址分配，但这时用不上的地址数就更多了。另外，网络越大，也就是广播域越大，自然在网络中的广播流量也可能越大，而划分子网后，网络更小了，广播域也更小了，产生广播风暴的可能性也就更小了。

或许你会说，这没什么问题啊，反正局域网**IP**地址不同公司是可以重复使用的，现有的网络地址数量一个公司无论如何也用不完，而

且又不用付费。这的确是这样的，但是这里还是有一个方面你没有考虑到，那就是路由效率的问题。你为每个局域网都选择一个标准的网络，那么每个局域网的网络路由条目中根据不同网络进行路由汇总，随着网络的增大，路由条目必须随之逐步增多，路由效率自然下降了。而采用**VLSM**进行子网划分后路由是可以汇总的，把由同一标准网络划分的多个子网路由条目汇总成一条标准网络的路由，可以使路由表条目大大减少，提高路由效率。

以上是针对局域网专用**IPv4**地址来说的，对于公网**IP**地址来说，采用**VLSM**所带来的好处就更明显了。试想一下，如果仅采用标准网络的分类，则每个**ISP**所能分配给企业用户的网络数就非常有限了，即使是最多的**C**类网络，全球可用的也只有2 097 152个（其中还有65 536个是局域网地址），这显然是不够的。这时你可能会说，这是**C**类网络总数，但是每个**C**类网络中还有254个地址啊，两者相乘就更多了。在此且先不说一个个来分配是否够用，事实上许多企业不可能只需要一个公网**IPv4**地址，而是有多个连续的**IPv4**地址，为多个互联网服务器进行分配。

我们知道，公网**IP**地址是在互联网中全球用户可以直接访问的（当然这要求用户已在公网中使用这个地址），除非做了特别的限制。而在标准网络中，每个网络的路由是相同的，如229.168.129.0/24这个网段的路由条目中就是包括这个网络地址，凡是在这个网络中的

地址均可通过这条路由到达。现假设A公司购买了一个227.126.128.10~227.126.128.20的C类公网地址段，而B公司购买了一个227.126.128.30~227.126.128.50的C类公网地址段，它们所购买的这些公网地址均用于一个为互联网用户提供服务的服务器网络中。

如果没有VLSM，A公司和B公司的局域网均只能通过配置227.126.128.0/24这条路由来实现内部局域网与服务器网络的互通。但这样一来，如果没有其他限制措施，则两家公司的这条路由都可能到达对方的服务器，这显然不行，很不安全，也不便于ISP对用户IPv4地址和域名服务的管理。但如果为每个用户分配整个网络的地址段，一方面用户不需要，也不可能花那么大资金来购买用不完的公网IPv4地址，另一方面ISP也不可能这样来分配，因为事实上IPv4地址资源非常有限，到了有钱也买不到的地步。如果采用VSLM，则可以很好地根据用户实际需要为每个用户设置一个单独的子网，这样就可以杜绝不同公司中公网地址网络的不安全性。

对于地址数是否够用的问题，相信不用我过多解释，全球有那么多企业或者个人用户都在申请购买公网IP地址，据悉目前IPv4可用的公网IP地址已全部分配完，没有余量了。对于公网IPv4地址说，采用VSLM同样可以像局域网中那样提高路由效率，特别是在那些大型的ISP广域中。

8.1.5 网络地址、主机地址和广播地址

在一些考试或者阅读一些技术资料时经常见到网络地址、主机地址和广播地址这几个术语，那么它们各代表什么意义呢？

1.网络地址、主机地址、广播地址的定义

网络地址是用来标识一个有类或无类网络的地址，是对应网络或子网的第一个IPv4地址，即主机ID部分全为0的IPv4地址；而广播地址则是一个有类或无类网络中的最后一个IPv4地址，即主机ID部分全为1的IPv4地址，可通过这个地址向对应网络或子网以广播方式发送数据包（也就是广播通信），让本地网络或子网的所有节点都可收到同一数据包。

在一个对应网络或子网中，除了网络地址和广播地址这两个一头、一尾的地址外，中间其他的所有地址都是主机地址，它是可直接分配给主机使用的IPv4地址。

如在192.168.1.0/24网络中（有关地址前缀表示形式将在下节介绍），网络地址为192.168.1.0/24，广播地址为192.168.1.255/24，其他254个IPv4地址都是主机地址。如192.168.1.0/26这样一个子网中，它的网络地址为192.168.1.0/26，而广播地址为192.168.1.63，而中间其他的

62个IPv4地址都是主机地址。有关子网划分的方法将在本章后面介绍。

说明 我们也经常听到单播、组播、广播这三个术语。这是IPv4协议根据所使用的IPv4地址通信用途来划分的三种方式：单播

（unicast）是指一台源IP主机仅与一台目的IP主机进行通信的方式。单播通信中所使用的IPv4地址就是单播地址，在A、B、C三类地址中（仅指IPv4）除了前面说到的广播地址，以及其他不能分配给主机使用或者保留使用的地址外，其他的全是单播地址，包括前面说的公网IP地址和私网IP地址。

组播（Multicast）是指一台源IP主机同时与网络中多台IP主机进行通信的方式，又称多播。组播通信所用的IPv4地址就是组播地址，D类IPv4地址就是组播地址。通过这类IPv4地址可以把一个数据包发送到多个指定的节点上。

广播（broadcast）是指一台源IP主机同时与本地网络或子网中所有其他节点进行通信的方式。广播通信所用的IPv4地址就是广播地址，就是指本节介绍的广播地址。

理解网络地址和广播地址要注意以下两个方面：

1) 某个或某几个8位组值为0或者255的IPv4地址不一定是网络地址或广播地址。不要一看到某个或者某几个8位组值全为0就认为是

网络地址，也不要一看到某个或某几个8位组值全为1就认为是广播地址，具体还是要看主机ID部分是否全为0，或1。如1.1.0.0/8这个地址就不是网络地址，因为它是一个标准的A类网络。主机ID部分应该是后面的全部3字节（24位），现在只有最后2字节的值全为0，而第二个字节的值为1。1.1.0.0/8的网络地址是1.0.0.0/8。同样，1.1.1.255/8、1.1.255.255/8也不是广播地址，原因同上，只是这里的主机ID部分（也就是最后3字节，共24位）应该全为1（每个字节转换成十进制为255），但这里只有最后2字节的值全为1。

对于无类网络来说更是这样，因为在一个无类网络中多个C类或B类有类网络都可以聚合成一个更大的网络，其中就可能包括许多某些8位组全为0或255的IPv4地址。

2) 网络地址和广播地址中不一定是整个8位组的值为0或255。也不是所有网络地址的主机ID都是整个字节的十进制值为0，所有网络地址的主机ID都是整个字节的十进制值为255。这仅适用于标准的有类网络，因为有类网络中，都是用整个字节来划分网络ID和主机ID的，但在无类网络中，并不是这样的。

在无类网络中，子网掩码长度是可变的，不是固定的多少个字节长度。这里涉及两个概念：子网划分和子网聚合。子网划分是在标准的有类网络基础上，通过向主机ID部分借位变成网络ID来实现的，相

反子网聚合是通过向网络ID借位变成主机ID来实现的。具体的子网划分和子网聚合方法将在本章后面介绍。

如要把192.168.1.0/24网络划分成四个子网，它就是通过向“主机ID”借2位来实现的，这4个子网分别是：192.168.1.0~192.168.1.63（192.168.1.0/26）、192.168.1.64~192.168.1.127（192.168.1.64/26）、192.168.1.128~192.168.1.191（192.168.1.128/26）、192.168.1.192~192.168.1.255（192.168.1.192/26）。它们都有自己对应的如下“网络地址”和“广播地址”：

□192.168.1.0/26子网的网络地址是192.168.1.0/26、广播地址是192.168.1.63/26；

□192.168.1.64/26子网的网络地址是192.168.1.64/26，广播地址是192.168.1.127/26；

□192.168.1.128/26子网的网络地址是192.168.1.128/26，广播地址为192.168.1.191/26；

□192.168.1.192/26子网的网络地址是192.168.1.192/26，广播地址是192.168.1.255/26。

由此可见，各子网的网络地址或广播地址并不是整个字节长度均为0或1，转换成十进制后也就是不一定是0或者255了。

2.广播地址类型

本来广播地址只有一种，就是每个标准有类网络中所说的最后那个IPv4地址，它可以在整个标准网络中进行广播（注意，路由器是不会转发广播包的）。但自从有了无类网络后，根据其广播范围的大小，IPv4广播地址就分出了以下几种：

（1）网络广播地址

网络广播地址是传统意义上的广播地址，也就是标准的有类网络中的广播地址。网络广播地址可以将数据包广播发送到本地有类网络内部所有节点上。IPv4路由器不转发目的IP地址为网络广播地址的广播数据包，也就是网络广播包只能在一个本地网络内部广播，不能被路由到其他网络中。

网络广播地址是通过将有类IPv4地址（A、B、C三类）中的所有主机ID部分全部设置为1而得到的，也就是每个主机ID的8位组均为255。例如，151.110.0.0是一个有类别地址的B类地址，则其网络广播地址为151.110.255.255。

（2）子网广播地址

子网广播地址是针对具体子网的广播地址。它仅可以将数据包发送到相应无类别子网内部的所有节点上。IPv4路由器也不转发目的IP

地址为子网广播地址的广播数据包，也就是子网广播包只能在一个子网内部广播，而不能被路由到其他子网中。

子网广播地址是通过将无类别地址的主机ID部分全部设置为1而得到的，因为在包含主机ID的8位组中还包括网络ID（严格地讲是子网ID），所以这时的广播地址中主机ID所对应的8位组值就不一定是255了。例如，192.168.1.0/26这个子网的广播地址是192.168.1.63，而不是192.168.1.255。

（3）全子网定向广播

全子网定向广播地址是针对一个标准有类网络划分的子网而言的，等于这个标准网络的广播地址。它可以将数据包以广播方式发送到由该标准网络划分的所有子网节点上。如前面所说的192.168.1.0/24标准有类网络通过向主机ID借两位的方法划分成192.168.1.0/26、192.168.1.64/26、192.168.1.128/26、192.168.1.192/26这四个子网，原来192.168.1.0/24标准有类网络的广播地址192.168.1.255就是这四个子网的全定向广播地址。这类广播包可以在以它划分的所有子网上广播，也就是IPv4路由器可以转发这类广播包。

（4）有限广播

有限广播地址是通过将IPv4地址的32个位全部设置为1（255.255.255.255）而形成的。在本地网络ID未知的情况下（如采用

DHCP服务自动分配IP地址的客户端），可以使用有限广播地址来进行本地网络或子网内部所有节点传送。IPv4节点通常仅在自动化配置过程（例如启动协议（BOOTP）或DHCP）中使用有限广播地址，如DHCP服务，DHCP客户端发出的请求报文和DHCP服务器返回的应答报文都是以255.255.255.255作为目的地址的，因为那时DHCP客户端还没有分配到具体的IP地址。

8.1.6 IPv4地址前缀表示形式

通过上节的学习我们已经知道，仅给出一个IPv4地址，我们仍然无法知道它所在的网络，也就无法仅通过这个IP地址来进行寻址和路由了。

传统的IPv4地址表示方法是在给出具体的IPV4地址的同时给出它所对应的子网掩码，如192.168.1.10、255.255.255.0等。这也是我们在为节点配置IP地址的同时要配置子网掩码的原因。但在日常书写时，这样显然比较麻烦，因为IPv4地址和子网掩码都有32位，转换成十进制后仍然各有四段，比较长。

为了能更加简便地书写，采取了一种比较简单的地址前缀表示形式，就是在一个IPv4地址（可以是网络地址，也可以是主机地址）后面先加上一个斜杠（/），然后在这个斜杠后面直接写出该地址所在网络的网络ID，或者子网掩码长度，因为网络ID长度决定了具体IPv4地址所属的网络。如192.168.1.10/24代表的是一个标准的C类网络IPv4地址，而10.1.0.10/8则代表了一个标准的A类网络IPv4地址。

当然，这种表示方式主要还不是为了标识标准的有类网络，因为标准的有类网络中，各类网络的网络ID长度和地址范围都是固定的，一看具体的IPv4地址中第一个8位组所属的取值段范围就知道属于哪类

网络，及所对应的网络ID长度了，根本不用标识。如A类IPv4地址第一个8位组取值的地址范围为1~126，网络ID长度固定为8位；B类IPv4地址第一个8位组取值的地址范围为128~191，网络ID长度固定为16位；C类IPv4地址第一个8位组取值的地址范围为192~223，网络ID长度固定为24位。

其实无论是网络ID，还子网掩码长度，最大的意义还是针对无类网络而言，因为这时它们是可变的，也没有A、B、C这样的分类了，无论第一个8位组属于哪个取值范围，都可以有各种网络ID、子网掩码长度。如有1.1.10.20/26这样的地址，也可以有192.168.1.10/8这样的地址，前者通过子网划分得到，而后者可以通过子网聚合得到。

与采用子网掩码表示形式一样，从一个IPv4地址表示形式中的地址前缀值可以看出这个IPv4地址所在的网络或子网，以及这个网络和子网的地址范围。如从131.107.0.0/18这个地址中可知它所对应的网络中有18位属于网络ID部分，有14位属于主机ID部分，可分配给主机的IPv4地址数就是 $2^{14} - 2 = 16382$ 。

8.1.7 几种特殊的IPv4地址

在我们平时进行IPv4地址配置和管理时，经常遇到这么几种特殊的IPv4地址，其中一种最重要的IPv4地址就是私网IP地址，也就是通常所说的局域网IP地址或者专用网络IP地址。还有像以0、127、169开头的IPv4地址。本节将做一个集中介绍。

1. 私网IP地址

为了提高IPv4地址的重复利用率，在设计IPv4地址时就专门在前面介绍的A、B、C这类IPv4地址中各自划分了一段专用于各组织局域网内部的地址段，这就是我们前面所说的私网IP地址（又称局域网专用IP地址或者专用网络地址）。私网IPv4地址在不同公司内部局域网中是可以重复使用的，且无须向IP地址管理机构申请、注册和购买的。在A、B、C类地址中各自划分的局域网专用地址段如下。

(1) 10.0.0.0/8 (10.0.0.0, 255.0.0.0)

在A类地址中，局域网内部节点可以使用10.0.0.1~10.255.255.254范围内有效单播IPv4地址。如果用地址前缀表示方式，则为10.0.0.0/8。在这样一个地址空间中有24个主机ID位，相当于最多可以

有 2^{24} （16 777 216）个IP地址（包括了网络地址和广播地址），满足了几乎所有大型规模局域网对IP地址的需求。

（2）172.16.0.0/12（172.16.0.0，255.240.0.0）

在B类地址中，局域网内部节点可以使用172.16.0.1~172.31.255.254范围内的有效IPv4单播地址。如果用地址前缀表示方式，则为172.16.0.0/12。针对具体的网络，仍是B类网络中所限制的16位主机ID个数，相当于最多可有 2^{16} （65 536）个IP地址（包括了网络地址和广播地址），满足了几乎所有中等规模局域网对IP地址的需求。

（3）192.168.0.0/16（192.168.0.0，255.255.0.0）

对于数量最多的小型局域网，其内部节点可以使用C类地址中的192.168.0.1~192.168.255.254范围内的有效IPv4单播地址。如果用地址前缀表示，则为192.168.0.0/16。针对具体的网络，仍是C类网络中所限制的8位主机ID个数，相当于最多可有 2^8 （256）个IP地址（包括网络地址和广播地址），满足了大多数小型规模局域网对IP地址的需求。

因为ICANN永远不会把专用地址空间内的IPv4地址分配给一个连接到互联网的组织，所以互联网路由器中也永远不会包含指向专用地址的路由。您也无法通过互联网连接到一个专用地址。正因为如此，

使用私网IPv4地址的主机必须将其互联网通信量请求发送到一个有效公用地址的应用层网关（例如一个代理服务器），或者通过一个网络地址转换（NAT）设备将此专用地址转换成一个有效的公网IPv4地址再与互联网连接。

2.169开头的IPv4地址

在运行Windows Server 2003或Windows XP的操作系统中，如果你采用的是自动IP地址分配方式，但本地网络中又没有部署用于自动IP地址分配的DHCP服务器，则这些系统主机就会自动获得一个以169开头的IP地址。这就是所谓的自动专用IP地址，其地址范围包括一个B类地址段——169.254.0.0/16，子网掩码为255.255.0.0。

3.127.0.0.1地址

我们在配置网络设备或者进行系统主机测试时，经常用到127.0.0.1这个地址。它不能分配给主机用，但它却可以用来进行各种地址管理操作，如Ping操作。其实这个地址有个专门的名称，即环回地址（Loopback Address），是主机IP堆栈内部的IPv4地址，主要用于网络软件测试以及本地机进程间通信，在IP网络中用来测试主机TCP/IP协议是否工作正常。无论什么程序，一旦使用回环地址发送数据，协议软件立即返回，不进行任何网络传输，所以执行Ping操作也

只在本机上进行环路测试，用来检测网卡是否工作正常（主要受操作系统中通信协议和驱动程序安装是否正确影响）。

4.0.0.0地址

严格说来，0.0.0.0不是一个真正意义上的IP地址，因为我们在介绍IPv4地址的分类时就说到，第一个8位组的值不能为0。但在实际的设备配置和网络中确实又要用到这样一个IPv4地址。其实它不是特指某个IPv4地址，在不同的情形中使用它有不同的含义。如我们在配置默认路由时就使用了0.0.0.0这个IPv4地址，它是代表所有不清楚路由的目的主机和目的网络。这里的“不清楚”是指在本机的路由表里没有特定路由表项指明如何到达这些主机或网络。在网络配置中，0.0.0.0又代表整个网络，即网络中的所有主机。

8.2 IPv4子网划分与聚合

在本章前面已说到，为了解决IPv4的不足，提高网络划分的灵活性，诞生了两种非常重要的技术，那就是VLSM（可变长子网掩码）和CIDR（无类别域间路由），把传统的标准IPv4有类网络演变成一个更为高效、更为实用的无类网络。VLSM用于IPv4子网的划分，也就是把一个大的网络划分成多个小的子网；而CIDR则用于IPv4子网的聚合，当然主要是指路由方面的聚合，也就是路由汇总。通过CIDR可以把多个小的子网路由条目汇总成一个大网络的路由条目，以减少路由器中路由条目的数量，提高路由效率。本节将分别对这两种技术和实现方法进行具体介绍。

8.2.1 VLSM子网划分的基本思想

子网这个概念其实是在有了VLSM技术后才有的，原来的标准网络中不存在子网的概念，因为那时每个标准网络都相互独立，互不包含。本节将具体介绍一下VLSM子网划分的实现机制和两种非常简单的划分方法。

通过VLSM实现子网划分的基本思想很简单，就是把原来标准网络IPv4地址中的网络ID部分向主机ID部分借位，把一部分原来属于主

机ID部分的位变成网络ID的一部分（通常称为“子网ID”），如图8-10所示。原来的网络ID+子网ID=新网络ID，子网ID长度决定了可以划分子网的数量。

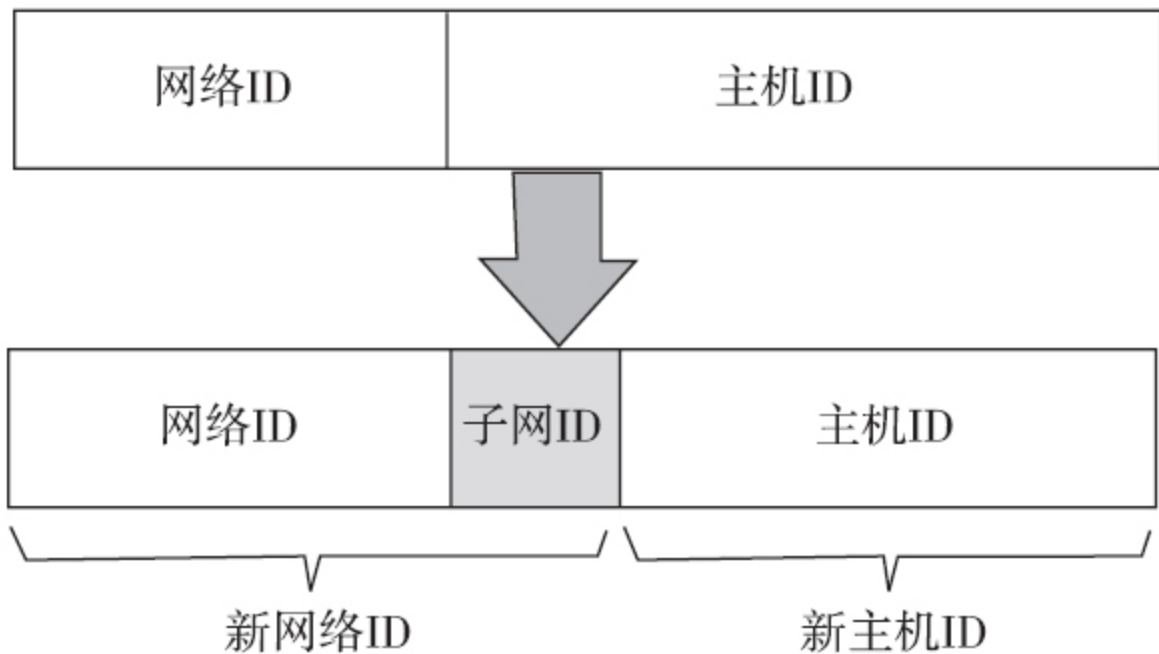


图 8-10 VLSM向“主机ID”借位示例

因为每一位都可以有0或1这两个值的两种选择，所以VLSM可以划分的子网数是 2^n 个（ n 为子网ID位数，也就是向原来主机ID借的位数），如借一位可划分成2个子网，借两位则可划分成4个子网，借三位则可以划分成8个子网，以此类推。不能划分成其他数量子网，如不能把一个网络仅划分成3、5、6、7、9之类个数的子网。但因为由同一个标准网络划分出来的每个子网的子网ID长度一样，这就决定所划分

出来的各子网新的网络ID长度也一样，所以各子网的子网掩码也一样。这点要特别注意。

这样一来，主机ID部分的长度变小了，也就是对应网络中所拥有的IPv4地址数减少了。当然VLSM的主要作用不是用来减小网络，更关键是它可以非常灵活地依据实际所需的地址数来调整所划分的子网大小。如原来一个标准的C类网络中有254个可以分配给节点的地址，现在公司中有6个部门，每个部门的人数不超过30人，现在想为每个部门分别配置一个子网。如果没有VLSM，则需要用6个C类标准网络。而有了VLSM后，仅用一个C类网络就可以划分出所需的6个子网，而且每个子网中可用的IPv4地址更贴近各部门的实际需求。这样就大大提高了IPv4的利用率，同时更便于网络管理，毕竟这些部门子网属于同一个大的C类网络。可以通过路由汇总功能提高路由效率，这对于ISP来说非常有用，因为可以大大节省所需的公网IP地址数。

经验之谈 VLSM只能划分相同大小的子网，也就是一个网络划分了子网后，各子网的IPv4地址数是相同的。要想使各子网大小不一样，则必须同时结合CIDR技术，把其中一些子网再聚合成一些稍大些的子网。另外，子网还可以进一步划分成更小的子网，这就是多级子网划分。具体划分方法将在下节通过具体的示例进行介绍。

有了VLSM，原来标准网络中的分类（如A、B、C类）方法就没有太大实际意义了，因为现在在网络IPv4地址规划中都会考虑使用

VLSM子网划分方法。原来标准网络中网络ID和主机ID部分都是固定不变的，而且是整字节长度，每一类网络有它特定的子网掩码。而重新划分子网后，**IPv4**地址的网络ID和主机ID部分都是可变长度的，子网掩码也不再限定于某一类网络之中，也就是同样一个子网掩码（如原来C类网络中的255.255.255.0）可以应用于原来各类网络中。

VLSM的设计目的就是想在每个子网上保留足够的主机数的同时，把一个标准的有类网络分成多个无类别的子网，提高原来标准网络中**IPv4**地址的有效利用率，也提高了网络组建时**IPv4**规划的灵活性，因为这样划分后，虽然各子网仍在一个大的标准网络之中，但彼此仍是二层隔离，必须通过三层才可以实现互通，这样就满足了许多企业用户不同部门网络隔离的需要。如一个只有200个左右用户的公司，现在想对几个部门组成几个独立的子网，如果仍按原来的标准网络分类，则必须使用几个C类网络，但每个子网只有几十个用户，所以每个子网所浪费的IP地址数非常多。如果采用了**VLSM**技术重新划分子网后，就可以仅使用一个C类网络来容纳多个部门子网了，而且每个部门都是独立的子网，就像以前多个C类网络一样。

8.2.2 全0子网与全1子网

全0子网和全1子网是针对进行子网划分后子网ID部分的取值而言的。全0子网代表的是对应子网的子网ID部分各位都是0，是第一个子网，而全1子网代表的是对应子网的子网ID部分各位都是1，是最后一个子网。

按照RFC950参考规定，划分子网后，只有 $n-2$ 个可用的子网（ n 表示总的子网数），第一个子网（也就是全0子网）和最后一个子网（也就是全1子网）不可用。但是在后来RFC1878规定中，该项规定已被废止了，现在的设备基本上都普遍支持RFC1878。现在来说，全0子网和全1子网事实上都是可以用的，因为有了CIDR，可以实现多子网路由聚合。有些比较老的设备还专门配备了一些命令来支持它们，如在Cisco路由器中可以通过“ip subnet-zero”命令配置来实现全0子网可用。

现在以一个示例来说明，为什么在RFC950要规定全0和全1子网不能用。假设我们有一个网络：192.168.0.0/24，现在要划分成两个子网，那么按照RFC950，应该使用/26（借用2个主机ID位，而不是/25只借用一个主机ID位），因为如果是/25则只划分了两个子网，而根据规定，第一个子网（全0子网）和最后一个子网（全1子网）都不能用，这样一来就没有可用的子网了。按/26划分方式可以得到两个可以

使用的子网192.168.0.64和192.168.0.128，第一个子网192.168.0.0和最后一个子网192.168.0.192不可用。

下面来分析一下各种网络的网络地址和广播地址。

□对于没划分子网前的192.168.0.0/24，网络地址是192.168.0.0，广播地址是192.168.0.255。

□对于重新划分子网后的192.168.0.0/26子网，网络地址是192.168.0.0，广播地址是192.168.0.63。

□对于重新划分子网后的192.168.0.64/26子网，网络地址是192.168.0.64，广播地址是192.168.0.127。

□对于重新划分子网后的192.168.0.128/26子网，网络地址是192.168.0.128，广播地址是192.168.0.191。

□对于重新划分子网后的192.168.0.192/26子网，网络地址是192.168.0.192，广播地址是192.168.0.255。

从上可以看出，重新按/26划分的第一个子网192.168.0.0/26的网络地址和没有划分子网时的192.168.0.0/24的网络地址是一样的，都为192.168.0.0。而重新划分后的最后一个子网192.168.0.192/26的广播地址和没有划分子网时的192.168.0.0/24的广播地址也是一样的，都为192.168.0.255。在CIDR正式使用以前，这样的重叠将导致极大的混

乱。比如，一个发往192.168.0.255的广播是发给主网络的还是子网的，就不清楚了。这就是为什么在当时不建议使用全0和全1子网的原因了。但在今天，**CIDR**已经非常普及了，通过地址前缀表示形式就可以很清楚地知道具体要发往哪个网络了，所以一般不需要再考虑这个问题。但本节仍以**RFC950**规定为标准进行介绍，确认需要使用第一个和最后一个子网时直接使用即可，子网的划分方法是一样的。

8.2.3 VLSM子网划分方法

许多读者朋友觉得子网划分好像非常难，他们总也搞不清楚。其实子网划分很简单。

在子网划分类试题或者具体工网络程中不外乎两种情形：

1) 已知要划分的子网数和最大地址数，求子网掩码、子网地址范围、网络地址和广播地址，其基本计算步骤如下：

a) 根据所需的子网数和最大地址数确定划分子网后的新网络ID长度和新主机ID长度；

b) 根据新主机ID长度确定子网划分后各子网的地址块大小，由此可进一步确定各子网的地址范围、网络地址和广播地址；

c) 根据下面的公式得出划分子网后各子网共同的子网掩码。

新子网掩码=原网络ID.新主机ID中各字节分别用256-各子地址块
(各字节的值之间以小圆点分隔)

2) 已知子网地址前缀或子网掩码，求子网地址范围、网络地址和广播地址。其基本计算步骤如下：

- a) 根据子网的地址前缀或子网掩码确定该子网地址块大小;
- b) 根据地址块大小, 可进一步确定该子网的地址范围、网络地址和广播地址;

从以上这两种情形的计算步骤中可以看出, 最关键的步骤都是要确定子网的地址块大小(也就是子网中的IPv4地址数, 也肯定是 2^n , n 代表“主机ID”位数)。知道了地址块大小就知道了包括主机ID部分各字节的子地址块大小, 因为总的地址块大小等于主机ID部分的各字节(也就是各个8位组)中2的对应二进制位数次方。如主机ID中包括IPv4地址第三个字节的后面4位和第四个字节, 则所划分的子网的两个子地址块分别是 2^4 (16) 和 2^8 (256)。

下面通过一个具体的示例来解释上面给出的新子网掩码计算公式。

现假设已知一个C类标准网络经子网划分后的地址块大小为64。经分析可知, 新主机ID部分仅为最后一个字节(因为一个完整字节的地址块大小256, 而64小于256), 又因原网络ID为255.255.255(因为原网络中一个C类网络)。根据上面的公式可以得出子网的子网掩码为255.255.255.(256-64), 即255.255.255.192。

再例如, 已知一个B类网络经子网划分后的地址块大小为1024。经分析可知, 新主机ID分布在最后两个字节中(因为这里的地址块大

小为1024，大于一个字节的地址块大小256，而小于两个字节的地址块大小65536）。再因 $1024=4\times 256$ （之所以要乘256，那是因为在这种情况下IPv4地址中最后一个字节肯定全部属于主机ID部分，它的大小就是256），所以可以得知这个子网地址中的两个子地址大小分别是4和256。又因原网络ID为255.255（因为原网络为一个B类网络），这样根据以上公式就可以很快得出子网的子网掩码为255.255.（256-4）。（256-256），即255.255.252.0。

下节将结合具体的示例对以上这两种情形下划分子网的方法进行具体介绍。

8.2.4 VLSM子网划分示例

本节针对上节介绍的两种典型情形介绍一些具体的VLSM子网划分示例，同时还将介绍一个二级子网划分的示例，以说明多级子网划分的方法。

1.已知所需子网数和最大地址数情形下的子网划分示例

【示例1】一公司原来使用的是192.168.1.0/24这个标准网络，现想为公司中每个部门（共6个）单独配置一个子网，其中最大一个部门要分配IPv4地址的数量不超过25个。求每个子网的子网掩码、地址范围、网络地址和广播地址。

这个示例是已知要划分的子网数和最大地址数。具体的划分方法很简单，先结合所需的子网数和最大地址数确定子网的网络ID和主机ID，然后根据主机ID确定子网的地址块大小，其他的就自然确定了。

下面是具体的计算步骤：

1) 本示例中，部门数为6个，但我们知道VLSM所划分的子网数都必须是 2^n ，可以看到与6最接近的就是划分成8个子网。而划分8个子网需要向主机ID借3位，这样子网的主机ID位数就仅有5位（8-3）了，则每个子网可使用的IPv4地址数为30个（ $2^5=32$ ，再减去每个子

网中不可分配给节点使用的网络地址和广播地址），恰好符合该公司中各部门的最大地址数要求。

2) 知道了新主机ID大小，也就是知道了各子网的地址块大小。
本示例中新主机ID共5位，大小为32 (2^5)。

3) 根据以上得出的地址块大小，再根据以上节介绍的公式可得出子网掩码为：原网络ID (255.255.255)。(256-32)，即
255.255.255.224。

4) 同样，知道了地址块大小，也就是知道了每个子网的地址范围。本示例的地址块大小为32，也就是把整个原来C类网络中256个地址按每块32地址划分成8等份（要注意的是，第一个地址要从0开始计算），得到的8个子网的地址范围如下：192.168.1.0~192.168.1.31、
192.168.1.32~192.168.1.63、192.168.1.64~192.168.1.95、
192.168.1.96~192.168.1.127、192.168.1.128~192.168.1.159、
192.168.1.160~192.168.1.191、192.168.1.192~192.168.1.223、
192.168.1.224~192.168.1.255。

各子网的地址范围知道了，自然就知道了它们的网络地址和广播地址了。每个子网的网络地址是对应地址段中的第一个地址，每个子网的广播地址是对应地址段中的最后一个地址。然后随便选择其中5个子网使用即可，最好是5个连续的子网，这样方便进行路由汇总。同时

如果你的设备不支持全0子网和全1子网的话，则不要选择第一个和最后一个子网。

【示例2】某公司想把原来使用的172.16.0.0/8标准网络分成12个子网，其中一个最大的子网节点数在4000个左右。求每个子网的子网掩码、地址范围、网络地址和广播地址。

这个示例同样是已知要划分的子网数和最大地址数，具体的划分方法很简单与上示例一样。下面是具体的计算步骤：

1) 本示例中要划分成12个子网，与12最接近的可划分子网数就是16，由此可见要向原来的主机ID借4位（ $2^4=16$ ），也就是第三个8位组的高4位。这样一来子网的主机ID位数就仅有12（ $16-4$ ）位了，则每个子网可使用的IPv4地址数为4094（ $2^{12}-2$ ）（同样是因为要减去每个子网中不可分配给节点使用的网络地址和广播地址）个，恰好符合该公司中最大子网地址数4000的要求。

2) 知道了新主机ID大小，也就是知道了各子网的地址块大小。本示例中新主机ID共12位，总地址块大小为4096（ 2^{12} ）。这个地址块是分布在第三个字节和第四个字节中，第三个字节有最低的4位，对应的子地址块大小为16，而第四个字节包括了全部的8位，对应的子地址块大小为256。

3) 根据以上得出的两个子地址块大小及上节介绍的公式可以得出子网掩码为：原网络ID (255.255) . (256-16) . (256-256) , 即 255.255.240.0 。

4) 同样知道了总地址块大小, 也就是知道了每个子网的地址范围。本示例的地址块大小为4096, 也就是把整个原来B类网络中65536个地址按每块4096地址划分成16等份, 得到的16个子网的地址范围如下: 172.16.0.0~192.16.15.255 (172.16.0.0/20) 、
172.16.16.0~172.16.31.255 (172.16.16.0/20) 、
172.16.32.0~172.16.47.255 (172.16.32.0/20) 、
172.16.48.0~172.16.63.255 (172.16.48.0/20) 、
172.16.64.0~172.16.79.255 (172.16.64.0/20) 、
172.16.80.0~172.16.95.255 (172.16.80.0/20) 、
172.16.96.0~172.16.111.255 (172.16.96.0/20) 、
172.16.112.0~172.16.127.225 (172.16.112.0/20) 、
172.16.128.0~172.16.143.225 (172.16.128.0/20) 、
172.16.144.0~172.16.159.225 (172.16.144.0/20) 、
172.16.160.0~172.16.175.225 (172.16.160.0/20) 、
172.16.176.0~172.16.191.225 (172.16.176.0/20) 、
172.16.192.0~172.16.207.225 (172.16.192.0/20) 、
172.16.208.0~172.16.223.225 (172.16.208.0/20) 、

172.16.224.0~172.16.239.225 (172.16.224.0/20) 、
172.16.240.0~172.16.255.225 (172.16.240.0/20) 。

同样，知道了各子网的地址范围，自然就知道了它们的网络地址和广播地址。然后随便选择其中12个子网使用即可，最好是12个连续的子网，这样方便进行路由汇总。同时如果你的设备不支持全0子网和全1子网的话，则不要选择第一个和最后一个子网。

【示例3】每个子网有不超过58个节点要分配IPv4地址，最合适的子网掩码是 () 。

A.255.255.255.192

B.255.255.255.248

C.255.255.255.224

D.255.255.255.240

这道题很简单，也是给出了子网中的最大地址数，那就是58个。由此可见，最恰当的方式是对一个C类标准网络进行子网划分，因为58小于一个字节的地址数256。而地址块中与58最接近的一个就是64（注意，地址块都是 2^n ， n 代表主机ID的位数），即 2^6 。这样一来立即可得出它的子网掩码为255.255.255.(256-64)，即255.255.255.192。

2.已知子网地址前缀或子网掩码的情形下的子网划分示例

这种情形下，子网掩码的相关计算就更容易了，可以直接通过地址前缀或子网掩码得出地址块大小，然后算出其他方面，如地址范围、网络地址和广播地址。

【示例4】IPv4地址为10.32.0.0，子网掩码为255.224.0.0的子网中最大可分配给主机的IPv4地址是什么。

这道题也是属于上节介绍的第二种情形，已知了子网的子网掩码。这时可根据子网掩码255.224.0.0很快得出该子网中的主机ID分布在IPv4地址后面三个字节中，各自的子地址块大小分别是：32（256-224）、256（256-0）、256（256-0）。

在这种主机ID分布在多个字节的情形中，起关键作用的就是包含主机ID中的第一个字节，如这里IPv4地址中第二个字节的“224”，相当于告知了该字节中的子地址块大小就是32（256-224），由此很快可以得出各子网的该字节取值范围就依次是0~31、32~63、64~95，……题中给出的10.32.0.0恰好在第二个子网中，它的IPv4地址段为10.32.0.0~10.63.255.255，因为广播地址不能分配给主机使用，所以该子网中可分配给主机使用的最大IPv4地址就是10.63.255.254。

【示例5】IPv4地址为202.112.14.137，子网掩码为255.255.255.224，所在子网的网络地址和广播地址各是什么。

这道题也是属于上节介绍的第二种情形，已知了子网的子网掩码。这时可根据子网掩码255.255.255.224得出其地址块大小为32（256-224）。因为它是一个C类标准网络地址，所以主机ID仅为IPV4地址中第四个字节。根据地址块大小32，可以把第四个字节的值划分为：0~31、32~63、64~95、96~127、128~159、.....由此可见202.112.14.137是在第五个子网中，网络地址就是202.112.14.128，广播地址是202.112.14.159。

3.多级子网划分示例

上面介绍的全是单级子网划分的示例，多级子网划的方法与单级子网划分的方法是一样，只不过是在子网的基础上进行再次划分。下面也举一个例子。

【示例6】把IPv4子网10.32.0.0/11划分成8个子网，求重新划分子网后的子网掩码和第3个子网的网络地址和广播地址。

这属于上节介绍的第二种情形，也就是已知了地址前缀，也就相当于已知了子网掩码。本示例是地址前缀为11，也就是原子网中的子网掩码中前11位全为1，后21位全为0，得到子网掩码为255.224.0.0。

现要对10.32.0.0/11子网进行二次子网划分，且划分的子网数是8个，也就是要再向主机ID借3位，由此可知新划分的子网中网络ID一共有14位（11+3），得到新子网掩码为255.252.0.0。

得出了新子网掩码后就可以得出属于主机ID部分的字节的子地址块大小（分别用256去减即可）。从新子网掩码为255.252.0.0可以得出，“主机ID”分布在第二个、第三个和第四个字节中，各自的子地址块大小分别为4（256-252）、256（256-0）、256（256-0）。同样关键是第一个属于“主机ID”的字节的子地址块大小，即4。所以10.32.0.0/11重新划分子网后的第二个字节的值是以4为单位进行划分的。从第一个新子网开始，第二个字节的值依次是32、35、39、43、47、51、55、59、63。题中所求的第三个子网就是第二个字节值为39的子网，它的地址段范围为10.39.0.0~10.43.255.255，其网络地址和子网地址自然就分别为10.39.0.0/14和10.43.255.255/14。

8.2.5 子网聚合方法及示例

在一些比较大的企业中，我们可能已经通过**VLSM**技术为它们的部门，或者分公司划分了许多子网。这样做的目的我们已经知道，一是可以提高**IPv4**地址的利用率，二是减小广播域（也就是“广播范围”的意思）大小，降低产生广播风暴的可能；三是可以提高网络的安全性。但是，这样进行子网划分后也会带来一些负面影响，如为了使得各子网的互通，在各路由器的路由表中就可能需要添加许多子网级的路由条目，使得整个网络的路由表变得更为复杂了，影响整个网络的路由性能。

这时我们就得使用另一项**IPv4**技术——**CIDR**（无类别域间路由）。**CIDR**可以把一些连续的多个小子网路由汇总成一条大网络的路由，这样通过这一条大网络路由条目就可实现这些子网间的路由。最终实现减少路由表中的路由条目和提高路由性能的目的。**CIDR**可以看成是**VLSM**子网划分的逆过程，是把多个小子网聚合成一个大的子网或标准有类网络，但必须同时借助**VLSM**来实现，所涉及的计算主要就是子网掩码，因为子网掩码与**IPv4**地址一起就可以确定路由中的具体目的网络。

其实子网聚合与前面介绍的子网划分实现机制上是一样的，就是通过改变网络的子网掩码长度来调整网络的大小。不同的只是子网划分是把网络ID向主机ID扩展（可以理解为向右走），也就是网络ID向主机ID借位，缩小网络。而子网聚合则相反，是把主机ID向网络ID扩展（可以理解为向左走），也就是主机ID向网络ID借位，扩大网络。当然，在实际的网络组建中，采用子网聚合的情况是极其少见的，CIDR路由汇总也是由路由器自动进行的。

既然子网聚合和子网划分都是通过借位来实现的，所以它们的本质还是一样的。在子网划分中向主机ID借 n 位就可以划分成 2^n 个大小相等的连续子网；而在这里介绍的子网聚合中是向网络ID借 n 位就可以聚合 2^n 个小相等的连续子网。

子网聚合的基本思想就是利用连续多个子网或标准网络的网络地址中相同的部分保留作为新网络的网络ID部分，不同的部分作为新网络的主机ID部分。然后由新网络的网络ID位数可以得出新网络的子网掩码。当然实际上还可以求出聚合后新网络的地址范围、网络地址和广播地址。下面也通过几个具体的示例进行介绍。

【示例1】把192.168.1.0/27、192.168.1.32/27、192.168.1.64/27、192.168.1.96/27这四个连续子网进行聚合。

把这四个连续子网的网络地址用二进制形式表示如下：

11000000.10101000.00000000.00000000

11000000.10101000.00010000.00000000

11000000.10101000.00011000.00000000

11000000.10101000.01100000.00000000

从以上可以看出，四个子网的网络地址中相同的部分就是用深颜色标注的这部分，即11000000.10101000.0，把这些位全部置1，即得出聚合网络的子网掩码为：255.255.128.0。

【示例2】聚合192.168.4.0/24、192.168.5.0/24、192.168.6.0/24、192.168.7.0/24这四个标准网络。

把这四个标准网络的网络地址转换成如下二进制形式：

11000000.10101000.00000100.00000000

11000000.10101000.00000101.00000000

11000000.10101000.00000110.00000000

11000000.10101000.00000111.00000000

结果得出相同部分共有22位，很快得出聚合后的子网掩码为255.255.252.0。

经验之谈 这里要特别注意的一点是，用于“子网聚合”的标准网络或子网必须是连续的，但连续情况不同，汇总的结果也不一样，如聚合第1~第4个子网，与聚合第2~第5个子网，虽然都是聚合四个子网，但结果就完全不一样，具体来讲比较复杂，所以也没有一个统一的方法可以很快得出聚合后网络的子网掩码。我们还是采取上述方法，把每个子网的网络地址用二进制列出来，然后找它完全相同的连续位，即可得到新网络的子网掩码。但总的来说有这样一个规律，就是新的子网掩码每增加一位，新的聚合子网数就是原来聚合子网数的2倍，反之亦然。如原来通过向“网络ID”借3位可以聚合了8个连续子网，现在假设再继续借一位，则可以聚合的连续子网数就是16个了。

【示例3】聚合192.168.5.0/24、192.168.6.0/24、192.168.7.0/24、192.168.8.0/24这四个标准网络。

同样把这四个标准网络的网络地址转换成如下二进制形式：

11000000.10101000.00000101.00000000

11000000.10101000.00000110.00000000

11000000.10101000.00000111.00000000

11000000.10101000.00001000.00000000

经过比较发现，这四个子网只有20位是完全相同的，也就是聚合后的子网掩码是20位，而不是上例所得到的22位，尽管它们聚合的都是四个连续网络。究其原因是因为给出的这四个网络不能聚合成一个网络，而必须再结合前、后连续的其他网络。其实本示例的聚合结果是对192.168.0.0/24~192.168.15.0/24共16个网络的聚合结果。

8.3 IPv4 NAT基础

由于IPv4地址空间太小，所以使得目前Internet正面临了两个关键问题：一是公网IP地址资源的匮乏，二是路由表的日益庞大，路由效率低。NAT（Network Address Translation，网络地址转换）技术与前面介绍的VLSM和CIDR技术一样，其出发点都是为了解决IPv4地址（主要是针对公网IPv4地址）不足的问题，提高现有IPv4地址利用率，但它们的实现机制不一样。前面介绍了VLSM和CIDR是通过调整子网掩码长度来充分利用现有IPv4地址的，而这里所介绍的NAT技术允许组织内部网络使用非全局可路由IP地址的用户通过地址转换为全局可路由的IP地址来访问Internet，以降低了对公网IP地址的需求。

NAT服务是运行在位于内、外网之间的路由设备上的，在内、外网用户之间通信时对数据包中的地址（可能是源地址，也可能是目的地址）进行转换。在当前仍是IPv4为主流协议的IP网络，NAT技术的应用非常广，因为它可以节约紧缺的公网IPv4地址。

8.3.1 NAT的主要应用

NAT服务工作在路由器上（同时可工作在三层交换机或防火墙上），通常用于连接两个网络，通过把内部网络中的多个私网IPv4地

址转换为合法的一个或多个公网IPv4地址，让使用非注册IPv4地址的私有IPv4网络全部或部分用户连接到Internet，或者允许Internet用户访问内部网络设备，如邮件服务器。

总体来说，NAT主要可以用于以下三种情形：

（1）无足够的公网IP地址可用时

当想要连接到Internet，但网络中并不是每个用户都有全局、合法的公网IP地址时，就可以使用NAT来解决。在连接内部网络和外部网络的边界路由器上配置NAT，在内部网络中多用户把数据包发送到外部网络之前把这些用户的数据包源IP地址中的内部本地地址转换为一个或者少数几个内部全局地址。这是NAT最基本的功能。目前大多数路由器共享上网方式都是采用这种NAT应用方案。

（2）重构网络IP地址部署

在你需要改变你的内部地址时，你可以使用NAT来进行改变，这样可以节省许多工作量。例如你原来采用的是一个C类网段（如192.168.1.0），后来由于用户数增加，导致IPv4地址不够用。按照常理是你可能要重新采用B类，甚至A类网段IP地址为网络中的用户重新分配。但这样一来，如果用户数较大，重新配置的工作量可能非常大。

此时我们可以采用NAT方式，让新增加的用户单独用另一个网段的IPv4地址（如192.168.2.0），然后再用一个支持NAT的设备与原来用户所在网段连接起来，通过NAT地址转换功能，只需要用一个原来网段的IPv4地址（如192.168.1.254）即可映射到新网段中所有用户的IPv4地址。这时新网段就相当于原来网段的一个设备，但可以实现两个网段处于同一网段的目的（因为新网段经过NAT地址转换后也为原来网段中的地址）。利用这种方法大大减少了重新配置的工作量。当然，这样做也存在一定的不便，如对新网段中的用户管理存在一定难度。因为新网段中所有用户是共享一个原来网段IP地址的。

（3）实现简单的TCP负载均衡

当你想要部署基本的TCP负载均衡时，你可以使用NAT把多个相同服务器的私网IPv4地址映射到一个全局公网IPv4地址，实现服务器的TCP负载均衡。这是NAT的TCP负载均衡功能。

8.3.2 与NAT相关的主要术语

图8-11所示是基本NAT应用网络结构和组成要素。首先起关键作用的当然是提供NAT服务的网络设备，通常是路由器（也可以是其他三层设备，如三层交换机和防火墙等）。它通过两个接口连接内、外两个网络。内部网络（**Inside Network**）就是用户当前所在网络，通常是指用户私有局域网，而外部网络（**Outside Network**）是指内部网络以外的其他所有网络，一般是指Internet。

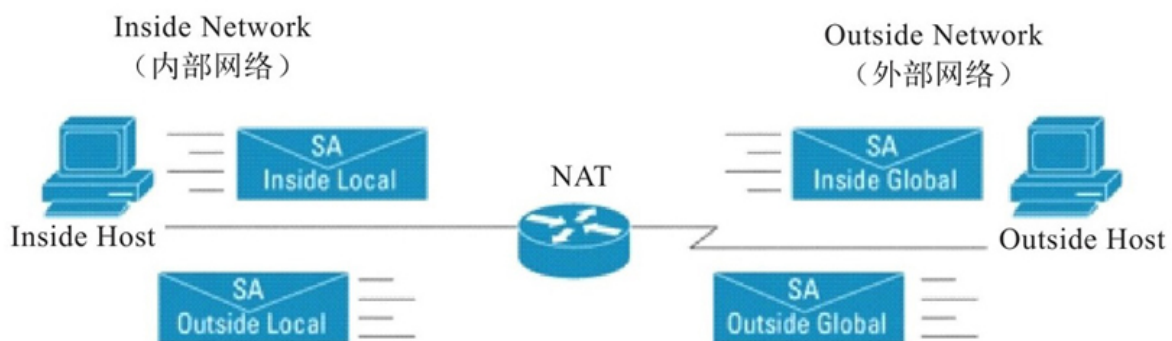


图 8-11 NAT应用拓扑结构的基本组成

下面是与NAT技术密切相关的几个术语描述，这对理解NAT技术工作原理非常重要。

(1) 内部网络（Internal Network）

内部网络通常是指一个边缘局域网，使用内部网络IPv4地址，尽管它也可以是注册的公网IP地址，但更多的是非注册的私网IP地址。使用非注册IPv4地址的所有计算机都必须使用NAT转换后再与其他网络进行通信；而使用注册的公网IPv4地址的主机如果是在由路由器隔离的内部局域网中，也是需要经过NAT转换后才能直接使用它进行访问的。所以总之，内部网络的IPv4地址都必须经过路由器的NAT转换才能访问外部网络。

（2）外部网络（External Network）

外部网络是除本地私有网络以外的所有其他网络，在NAT应用中，Internet是最常见的外部网络。当然，外部网络也可以是其他私有网络，如两个局域网通过路由器相连的情况下。所以外部网络上的用户使用IP地址同样既可以是注册的，也可以是非注册的。

经验之谈 其实内部网络与外部网络只是相对定义的，它是相对你当前所在的网络的而言的，如果你换到另一方网络去配置（在路由器连接的是两个局域网的情况下），这时内、外部网络的角色就要互换了。

在NAT中，除了以上两个网络类型的定义外，还依据IPv4地址是在专用网络还是在公用网络，以及通过方向是流入还是流出有不同的定义。

(3) 本地地址 (Local address)

在IPv4地址中，可以根据IPv4地址的作用范围分为本地地址 (Local address) 和全局地址 (Global address) 两大类。

本地地址是本地网络（可以是内部网络，也可以是外部网络）内部使用的IP地址，仅在本地网络有效，不能直接用来访问外部网络的IP地址，不可路由，这就是“本地”两字的含义所在。前面说了，内部网络中既可以使用未注册的私有网络IP地址，又可以使用公网注册的IP地址，所以，本地地址既可以是通常所见的私网使用的非注册IP地址，也可以是在公网中使用的注册IP地址，但通常是指私网IP地址。

因为NAT连接了内、外两个网络，所以在本地地址中又有两类，一类是用于内部网络的内部本地地址 (Inside Local Addresses)，一类是用于外部网络的外部本地地址 (Outside Local Addresses)。

内部本地地址 (Inside local address)

内部本地地址是指分配给内部网络主机的IP地址。这个IP地址是计算机操作系统，或诸如DHCP之类的服务进行分配的，但既可以是仅限于内部局域网使用的私有非注册IP地址，也可以是由ISP统一分配的注册IP地址。它们都是在内部网络使用的，通常是指非注册IP地址。

外部本地地址 (Outside local address)

这是本地地址的另一种，与内部本地地址性质一样，是为外部网络主机分配的本地网络IP地址，是外部网络主机对内部网络用户呈现的IP地址。这个IP地址是计算机操作系统或诸如DHCP之类的服务进行分配的，但也既可以是仅限于内部局域网使用的私有非注册IP地址，也可以是由ISP统一分配的注册IP地址。通常是指专用网络使用的非注册IP地址。

经验之谈 这里所说的内部本地地址和外部本地地址不是路由器上连接内、外部网络的接口的IP地址，而是在内、外部网络中主机分配给的IP地址。这一点一定要搞清楚。

（4）全局地址（Global address）

全局地址是与本地地址相对应的IP地址，它是内、外部网络本地地址转换后的IP地址的，是可路由的。全局地址其实是指在内、外网络中间架设的一个过渡性网络的IP地址。内、外部网络相互通信都需要先把对应的本地地址转换成对应的全局地址才能与对方网络进行通信。考虑到NAT路由器既有与像互联网这样连接局域网的情形，又有连接两个局域网的情形，所以全局地址也既可以是可供注册的公网IP地址，也可以是非注册的私网IP地址。但NAT的应用主要是内部局域网与Internet的连接，所以外部网络一般是Internet，这样，全局地址也就一般是公网注册IP地址了。

全局地址也分为两类：一是用于转换内部本地地址的内部全局地址（**Inside global address**），另一类是用于转换外部本地地址的外部全局地址（**Outside global address**）。

内部全局地址（**Inside global address**）

内部全局地址是内部网络主机对外部网络用户呈现的**IP**地址（可以是分配给路由器连接外部网络接口的**IP**地址），是内部本地地址转换后的地址。通常它是由**ISP**分配给企业用户内部网络使用的注册**IP**地址，但也可以是服务提供商分配、在本地注册的私网**IP**地址。通过**NAT**转换后，对于外界网络来说，它们扮演的是一个或多个内部本地地址**IP**地址，以便与外部网络通信。

外部全局地址（**Outside global address**）

外部全局地址是外部网络主机分配到的外部网络**IP**地址。它通常是由**ISP**分配给企业用户内部网络使用的注册**IP**地址，但也可以是服务提供商分配、注册的私网**IP**地址。

8.3.3 NAT地址基本转换原理

在私有网络中的计算机是使用内部本地地址（**Inside Local Addresses**）进行通信的，当它们需要与外部网络进行通信时，就需要为他们配置内部全局地址（**Inside Global Addresses**）。

总体来说，NAT进行地址转换的过程就是本地地址与全局地址之间的转换过程，无论数据包是从内部网络发往外部网络，还是从外部网络发往内部网络。不同的只是本地地址和全局地址所对应的网络不同，以及数据包重新封装的源和目的地址不同。具体如图8-12所示。

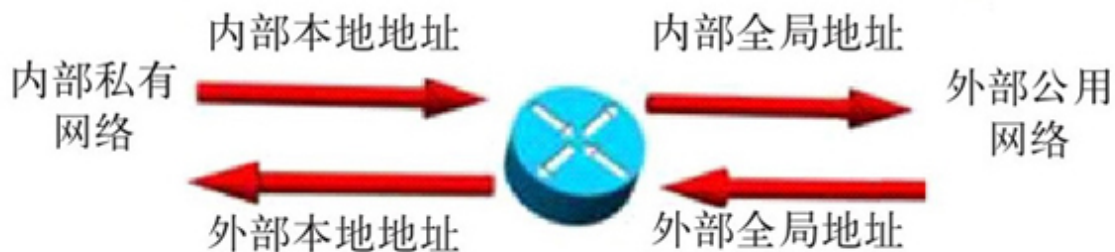


图 8-12 NAT基本地址转换原理

在以上转换过程中，当数据包还在内部网络位置时有一个作为源地址的内部本地地址和一个作为目的地址的外部本地地址。此数据包首先发往路由器连接内部网络的接口中。当数据包被转发到外部网络时，数据包的源地址就会转变为内部全局地址，而目的地址被转变为外部全局地址，如图8-13所示。也就是把数据包的所有源和目的地址全

部由本地地址转换为全局地址。这个过程是通过NAT中的本地地址与全局地址映射条目来实现的，所以事先要在NAT路由器上配置这样的映射条目。

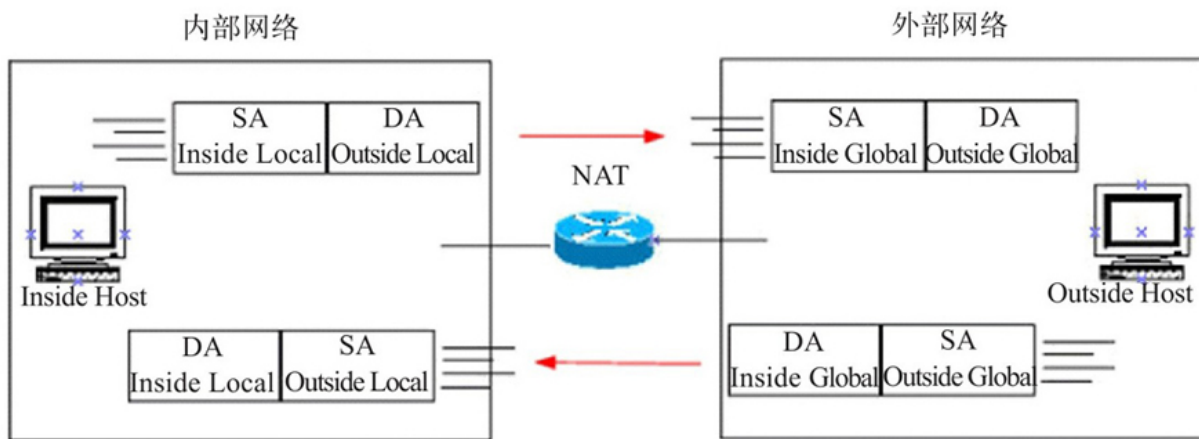


图 8-13 NAT的详细地址转换原理

相反，当数据包是从外部网络位置发来，并且仍位于外部网络中时，它的源地址就是外部全局地址，目的地址就是内部全局地址。相当于由内部网络向外部网络发送数据包时，外部网络主机接收到的数据包中的源地址和目的地址的互换。而当数据包被路由器转发到本地网络时，源地址被转变为外部本地地址，目的地址被转变为内部本地地址，也相当于由内部网络向外部网络发送数据包时，内部网络主机发送的数据包中的源地址和目的地址的互换。

从以上分析可以看出，由内部网络向外部网络转换时，以及由外部网络向内部网络转换时，IP报文中的源IP地址（SA）和目的IP地址（DA）在内部本地（Inside Local）地址与内部全局（Inside Global）地

址，外部本地（**Outside Local**）地址与外部全局（**Outside Global**）地址之间的转换方式。

8.3.4 NAT类型

在NAT路由器中虽然可以配置多种形式的NAT地址转换，但总体来说就是两种：静态NAT和动态NAT。在动态NAT中又有一种特殊的NAT形式，那就是重载（Overloading）NAT，也就是通常所说的PAT（端口地址转换）。而还有一种针对内部网络、也使用公网注册IP地址的这种特定网络配置情形的NAT转换形式，也就是后面将要介绍的重叠（Overlapping）NAT。它既可以静态NAT方式部署，也可以动态NAT方式部署。下面具体介绍这几种NAT类型。

1.静态NAT（Static NAT）

静态NAT是把非注册IP地址（如本地局域网IP地址）一对一地映射到公网注册IP地址（如互联网IP地址）。这在网络设备需要以互联网IP地址访问外网时特别有用。但一定要注意，这里仅列举了单一的正方向的IP地址转换的例子，实际上还可以是反方向或者双方向进行IP地址转换的，下面的动态NAT和复用NAT也一样可以有正向、反向、或者双向转换方式。

图8-14所示的是一个静态NAT应用示例（注意箭头方向）。路由器内、外两个网络，左侧内部网络中的192.168.32.10、192.168.32.12和192.168.32.15这三台主机使用的私有网络IP地址，通过路由器的NAT功

能最终对应转换成213.18.123.110、213.18.123.111和213.18.123.112这三个公网IP地址，让对方看到的也是这三个公网IP地址。

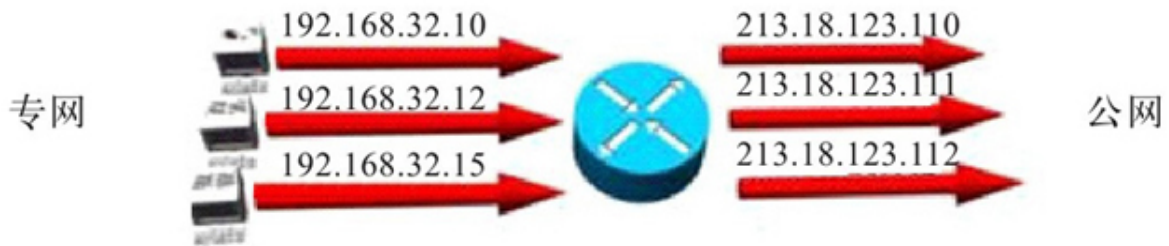


图 8-14 静态NAT应用示例

2.动态NAT (Dynamic NAT)

动态NAT可把一个非注册IP地址动态地映射到一个注册IP地址池中的一个地址。具体是哪两组IP地址之间的映射关系，还要看所配置的具体公用IP地址池和通信时间。但最终非注册IP地址与注册IP地址还是一对一地地进行映射的。

图8-15所示是一个动态NAT应用的示例。内网中的三个IP地址与一个范围为213.18.123.100~213.18.123.150的公网IP地址池进行动态映射。最终的结果是，192.168.32.10映射为213.18.123.116，192.168.32.12映射为213.18.123.112，而192.168.32.15映射为213.18.123.125，.....



图 8-15 动态NAT应用示例

3.重载或复用NAT（Overloading NAT）

重载NAT是动态NAT的一种形式。它通过与IP地址的不同端口组合，把多个非注册IP地址映射到一个注册IP地址，也就是通常所说的PAT（Port Address Translation，端口地址转换）。通过PAT，数千内网用户都可以仅通过一个公网IP地址访问Internet。

图8-16所示是一个重载NAT的应用示例。示例中本地网络中的所有用户通过路由器访问公用网络时，都将映射成同一个公网IP地址——213.18.123.100，只是所使用的端口不同而已（分别为101、102、103号端口）。这对于公网IP地址比较紧张，而内网中又部署了多种应用服务器时特别有用，可以通过一个公网IP地址配置多个应用服务器。



图 8-16 重载NAT应用示例

4.重叠（Overlapping）NAT

当内部网络中使用的IP地址是外部网络中注册的IP地址时（也就是内、外部网络使用的是相同IP地址段时），路由器需要维护一张查询表，以便截取来自内部网络的数据包，并用外部网络中没有使用的注册IP地址进行替换。但要注意，NAT路由器必须转换内部网络本地地址为注册的唯一（也就是没有被使用的）IP地址，同时也必须转换外部网络本地地址为内部网络中唯一的IP地址。这既可以通过静态NAT来实现，又可以通过使用DNS和动态NAT来实现。

经验之谈 重叠NAT主要应用于使用公网IP地址的内部网络服务器的地址转换，这样可以隐藏内部网络服务器的真实公网IP地址，有利于保护服务器的安全。因为经过NAT转换后，内部网络服务器对外呈现的是另一个公网IP地址，不是真实的服务器IP地址。

图8-17所示为一个重叠NAT的应用示例。在内部网络中使用的IP地址段为237.16.32.xx（也是注册的公网IP地址），而外部网络主机也使用这个地址段。这时，NAT路由器需要转换内部网络中的本地IP地址

（如237.16.32.10）为一个区别于外部网络的另一网段的全球地址（如213.18.123.103），以避免与外部网络中的用户造成潜在的冲突。同时在由外部网络向内部网络发送数据时，NAT路由器也将转换内部全球地址（如213.18.123.103）为内部本地地址（如237.16.32.10）。



图 8-17 重叠NAT应用示例

8.4 IPv6地址基础

前面介绍了IPv4协议的编址方案，与之相比，IPv6的编址方案要复杂许多。IPv6与IPv4协议相比，最大的特征就是地址空间增大了许多倍，因为IPv6地址的位数由原来的IPv4地址的32位激增到了128位，相当于IPv4地址空间的4倍。与IPv4地址通常是采用圆点分隔的十进制表示类似，IPv6是由使用由冒号分隔的十六进制数表示（每段4位十六进制数，相当于16位二进制数，如图8-18所示），且在IPv6地址中，十六进制数对大小写不敏感，也就是大、小写任意。由于地址空间扩展了3倍，所以从目前看来，可以说是彻底解决了IPv4地址缺乏的问题，据悉IPv6地址全部使用后，可以使全球每个人可以拥有平均10多个地址。

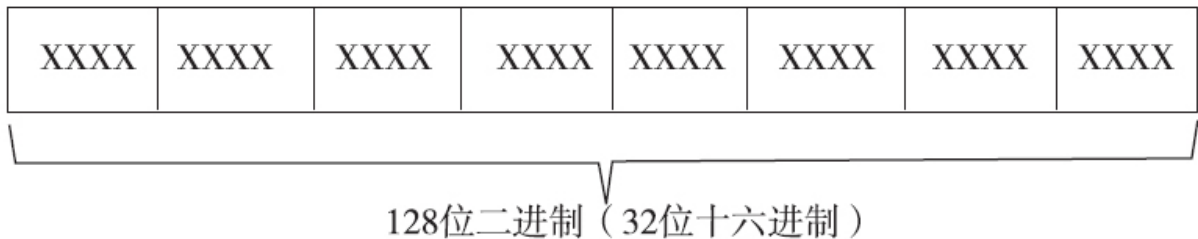


图 8-18 128位IPv6地址格式

8.4.1 IPv6地址表示形式

IPv6地址长度4倍于IPv4地址，表达起来的复杂程度也是IPv4地址的4倍。IPv6地址的基本表达方式是x:x:x:x:x:x:x:x，其中的一个“x”是一个4个十六进制整数，每个十六进制数代表4位二进制，每个IPv6地址包括8个整数（也就是8个地址块），这样一来就得出，IPv6总的位数为128位（ $4 \times 4 \times 8$ ）个二进制。

在IPv6协议中，一个128位地址空间中，允许 2^{128} ，即340 282 366 920 938 463 463 374 607 431 768 211 456（或约等于 3.4×10^{38} ）个（也就是43个43亿）可能的地址。而在IPv4协议中，一个32位地址空间允许 2^{32} 即4 294 967 296个（约43亿）可能的地址。由此可见，IPv6的地址空间是IPv4的 2^{96} 倍。与IPv4相比，地址空间不是很小的扩展，而是一个天文数级别的扩展。

1.基本的“冒号十六进制”表示形式

在我们的日常使用中，IPv4地址是每8位以点分十进制形式（每个8位组中间是以小圆点分隔的）来表示的，如192.168.10.12。对于IPv6，128位的地址则是按每16位（是IPv4中每块位数的2倍）一组来分隔的，每个16位块转换成4个十六进制数（注意不是十进制了），这样就一共划分成了8块（也是IPv4块数的2倍）。且相邻的16位块是以半角冒号（不再是IPv4地址中的小圆点）隔开。所以，生成的IPv6地址表示形式称为冒号十六进制表示形式。

下面是一个二进制形式的IPv6地址（共128位）示例：

```
0011111111111110:0010100100000000:1101000000000101:000000000
0000000:0000001010101010:0000000011111111:111111000101000:10011
10001011010
```

在日常的使用中，IPv6地址不是以二进制来表示的，而是以十六进制表示的（主要也是为了书写方便，如果也用十进制来表示的话，这个十进制数就会很大，最大数为65535）。把每个16位块都转换成十六进制，相邻的块用半角冒号隔开即可。以上示例转换的结果为（注意有8块组成）：

```
3FFE:2900:D005:0000:02AA:00FF:FE28:9C5A
```

2.前导零省略表示形式

从上面的十六进制IPv6地址来看，仍然相当长，仍不便于书写，书写时也容易出错。为了进一步简化IPv6地址，在IPv6地址编址方案中新增了一个规定，那就是如果在一个十六进制整数（也就是一个地址块）中高位为0，可以省略。如

3FFE:2900:D005:0000:02AA:00FF:FE28:9C5A中的02AA，则就可以写成2AA；00FF可以直接写成FF。如果整个地址块都是0，则只需要写一个0即可，如上面示例中的0000，就可以写成0。这样一来，这个示例就可以进一步简化写成：

3FFF:2900:D005:0:2AA:FF:FE28:9C5A

又如1080:0000:0000:0000:0008:0810:213C:123A这样的IPv6地址，采用以上方案后就可直接写成1080:0:0:0:8:810:213C:123A。

3.压缩零表示形式

虽然通过前面介绍的前导零省略法已对IPv6地址表示形式进行了简化，但由于IPv6地址规定有8段、128位之多，即使按上述方法把“0”的写法简化了，在各16位地址块中的“0”的位数一多的情况下也会显得有点麻烦，或者意义不大。为了进一步简化，又导入了双冒号的规则。即可以用双冒号置换地址中连续的、值为“0”的16位地址块（转换为十六进制后就是4位了，下同），也就是通常所说的双冒号置换法。

如上面的1080:0:0:0:8:810:213C:123A地址中间有3个连续的地址块为0，此时可将3个连续为的0地址块用一个“:”置换（包括原来连续地址块间的冒号）。这样一来，上面示例就可以进一步简写成：

1080::8:810:213C:123A，更加简化了。

例如，您可以将FE80:0:0:0:2AA:FF:FE9A:4CA2的IPv6地址压缩成FE80::2AA:FF:FE9A:4CA2；可以将FF02:0:0:0:0:0:0:2的IPv6地址压缩成FF02::2；可以将2000:0:0:0:0:0:0:1的IPv6地址表示为2000::1。
0:0:0:0:0:0:0:1等价于::1； 0:0:0:0:0:0:0:0等价于::。

注意“双冒号置换法”中的双冒号所代表的值为0的连续16位地址块个数不限，但是双冒号在一个地址中只能使用一次，也就是说不能重复使用它来替换多个分隔的、连续地址块中的多个0。例如有这样一个IPv6地址：0:0:0:AB98:123C:0:0:0。按上述双冒号置换原则，可缩写成::AB98:123C:0:0:0或0:0:0:AB98:123C::，但却不能写成::BA98:7654::，因为这样在一个地址中两次使用冒号置换后，您就无法确定每个“::”代表多少位零。

另外，“双冒号置换法”中的双冒号只能置换连续的值为0的16位地址块，不能置换非连续的值为0的多个16位地址块。如FE80:0:2AA:0:FF:0:FE9A:4CA2，就不能用双冒号置换了。

还有，只能使用压缩零来压缩以冒号十六进制表示形式表示的一个连续的，值为0的16位块，不能对16位块的一部分的0进行零压缩。例如，您不能将FF02:30:0:0:0:0:0:5表示为FF02:3::5，因为在这个书写格式中，把作为16位块30中一部分的0也省略了。

在压缩后的地址中要确定其中的“::”（双冒号）代表了多少个0，可以计算压缩地址中的块数，用8减去此数，然后将结果乘以16就是了。例如，地址FF02::2中显示只有两个地址块（“FF02”块和“2”块），这意味着其他6个16位块（总共是8个地址块）已被压缩，也就是说这个双冒号代表了其中6个全为0的16位地址块，一共是 $6 \times 16 \text{位} = 96 \text{位}$ 。

最后总结一下IPv6地址的书写规则:

1) 16位地址块中的前导0可以省略, 如果16位全为0, 可以只写一个0;

2) 当IPv6地址中有多个连续的, 值为0的16位地址块时, 可以用1个双冒号转换这些连续的0, 但双冒号在一个IPv6地址中只能出现一次, 也就是在一个IPv6地址只能用一个双冒号转换一个连续的、值为0的16位地址块。

3) 不能用双冒号转换属于16位地址块中一部分的0, 即使是地址块中的最后一个16进制数0。

8.4.2 IPv6地址中的二进制数与十六进制转换

因为在IPv6地址中使用的是二进制或者十六进制，而不再是IPv4地址中的二进制或者十进制，所以这就涉及了二进制与十六进制之间的转换问题。这在第1章已介绍过。在此仅就IPv6地址中的二进制与十六进制转换方法做一个简单介绍。其实就是给出了4位二进制所对应的十六进制，如表8-2所示（其中的十进制主要是为了方便对照）。

表 8-2 十进制、十六进制和二进制之间的转换对照表

十进制	十六进制	二进制	十进制	十六进制	二进制
0	0	0	8	8	1000
1	1	1	9	9	1001
2	2	10	10	A	1010
3	3	11	11	B	1011
4	4	100	12	C	1100
5	5	101	13	D	1101
6	6	110	14	E	1110
7	7	111	15	F	1111

要将十六进制数转换成二进制数，应将每个十六进制数字转换成对等的4位二进制数字。例如，要将十六进制数03D8转换成二进制，应将每个十六进制数字（0、3、D和8）分别转换成二进制。这样，0x03D8就是0000 0011 1101 1000或0000001111011000。

要将二进制数转换成十六进制数，应从低序位开始将二进制数分割成包含4位的块，最后不足4位时在前面以“0”补足。然后，再将每个

包含4位的块转换成与其相等的十六进制形式。例如，要将二进制数110000110101110转换成十六进制，应首先将整个数分割成4位一块的数字块，即0110 0001 1010 1110（第一个0是补的）。然后，将每块转换成十六进制数字，即61AE。

8.5 IPv6地址类型

IPv6协议主要定义了三种地址类型：单播地址（Unicast Address）、组播地址（Multicast Address）和任播地址（Anycast Address）。与原来在IPv4地址相比，新增了“任播地址”类型，取消了原来IPv4地址中的广播地址，因为在IPv6中的广播功能是通过组播来完成的。

□单播地址：用来唯一标识一个接口，类似于IPv4中的单播地址。发送到单播地址的数据报文将被传送给此地址所标识的一个接口。

□组播地址：用来标识一组接口（通常这组接口属于不同的节点），类似于IPv4中的组播地址。发送到组播地址的数据报文被传送给此地址所标识的所有接口。

□任播地址：用来标识一组接口（通常这组接口属于不同的节点）。发送到任播地址的数据报文被传送给此地址所标识的一组接口中距离源节点最近（根据使用的路由协议进行度量）的一个接口。

IPv6地址类型是由地址前缀部分来确定，主要地址类型与地址前缀的对应关系如表8-3所示。有关IPv6地址前缀将在本章后面介绍。

表 8-3 IPv6 地址类型与地址前缀的对应关系

地址类型		地址前缀（二进制）	IPv6 前缀标识
单播地址	未指定地址	00...0 (128 bits)	::/128
	环回地址	00...1 (128 bits)	::1/128
	链路本地地址	1111111010	FE80::/10
	站点本地地址	1111111011	FEC0::/10
	全球单播地址	其他形式	—
组播地址		11111111	FF00::/8
任播地址		从单播地址空间中进行分配，使用单播地址的格式	

8.5.1 IPv6单播地址

IPv6单播地址与IPv4单播地址一样，都只标识了一个接口。为了适应负载平衡系统，RFC 3513允许多个接口使用同一个地址，只要这些接口作为主机上实现的IPv6的单个接口出现。整个IPv6单播地址包括以下五个类型：全局单播地址、链路本地地址、站点本地地址、特殊地址、兼容性地址。下面分别予以介绍。

1.全局单播地址

全局单播地址等同于IPv4中的公网地址，可以在IPv6 Internet上进行全局路由和访问。这种地址类型允许路由前缀的聚合，从而限制了全球路由表项的数量。RFC3513给出了新的IPv6全局单播地址通用格式，如图8-19所示。

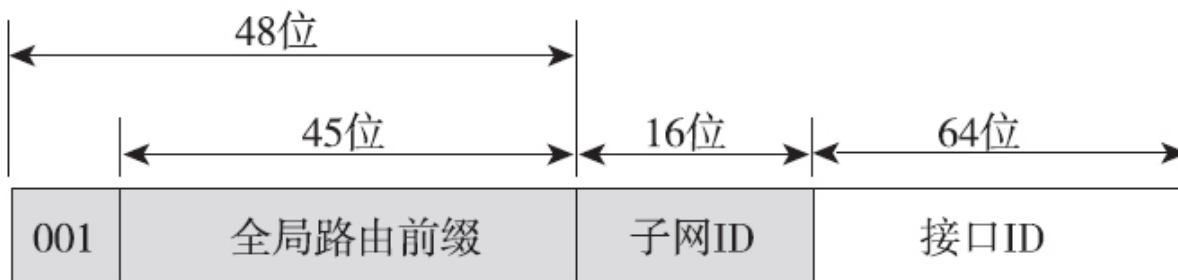


图 8-19 IPv6全局单播地址结构

全局单播地址包含四个组成部分说明如下：

□001：这是当前可分配的全局地址的地址前缀，此部分可表示为2000::/3。

□全局路由前缀：指示特定组织的站点的全局路由前缀。前面三个固定位与本部分的45位全局路由前缀一起组成48位的站点前缀，将其分配给组织的单个站点。分配了此前缀之后，IPv6 Internet上的路由器将与该48位前缀匹配的IPv6通信转发到组织站点的路由器。

□子网ID：用于在组织站点中标识子网，占16位。用户可以在站点内使用这16位来创建65 536个子网或多个级别的寻址层次结构以及有效的路由基础结构。

□接口ID：指示站点内特定子网上的接口，占64位，这部分对应各接口的64位IPv6MAC地址，由接口自动分配。其实也可不用MAC地址来指定接口ID，人为编制也可以，下同。

图8-20所示为全局单播地址内的各组成部分组成的三个级别结构。公用拓扑是提供对IPv6 Internet的访问的大型和小型ISP的集合；站点拓扑是组织站点内的子网的集合；接口标识符用于标识组织站点内的子网上的特定接口。

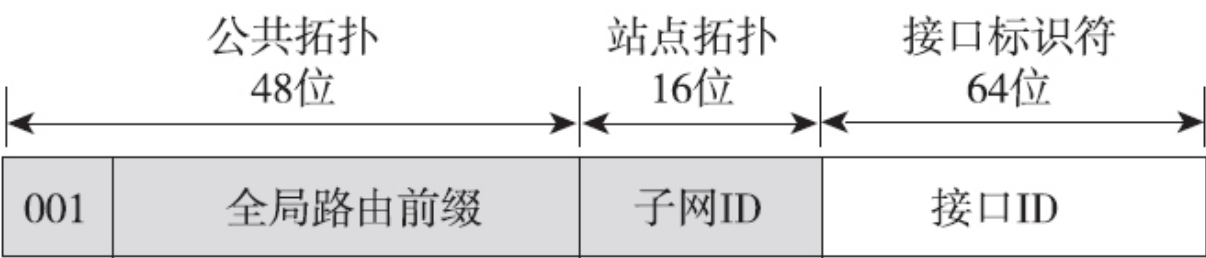


图 8-20 全局单播地址三个级别的结构

接口ID是来自接口的链路层地址（MAC地址），但这里的MAC地址不是我们现在IPv4中使用48位MAC地址，而是在IPv6中扩展后的64位MAC地址——EUI-64 MAC。在一个结点上的多个接口上可以使用相同的接口ID，只要他们连接的是不同的子网。

对于所有单播地址，除了以“000”开头的这些地址外，接口ID（也就是IPv6 MAC地址）必须是64位长，而且要符合IPv6改进的EUI-64格式。64位EUI-64地址是由电气和电子工程师协会（IEEE）定义的。EUI-64地址或者被指派到网络适配器，或者从IEEE 802地址派生得到。

IEEE EUI-64地址代表网络接口寻址的新标准。其中的“公司ID”部分仍然是24位长度，但“扩展ID”部分是40位，从而为网络适配器制造

商创建了更大的地址空间，如图8-21所示。



图 8-21 IPv6中的EUI-64地址结构

IANA负责IPv6地址空间的分配。目前IANA从整个全局单播地址空间（格式前缀为001）中取2001::/16进行分配。IANA指定的注册机构把2001:0600::/23到2001:0800::/23段IPv6地址分配给欧洲和中东（包括中国）。这些注册机构再从IANA得到的地址空间分配/32前缀给IPv6 ISP，IPv6 ISP再从/32前缀中分配/48前缀给每个客户。/48前缀的地址空间还可以进一步分为/64前缀的子网。这样每个客户最大可以有65535个子网。

2.IPv6本地单播地址

在IPv6中，本地单播地址就是指本地网络使用的单播地址，也就是IPV4地址中经常所说的局域网专用地址。本地单播地址又有两种，分别是链路本地地址和站点本地地址。每个接口上至少要有一个链路本地单播地址，另外还可分配任何类型（单播、任播和组播）或范围的IPv6地址。

(1) 链路本地地址

链路本地地址仅用于单个链路（注意，这里的“链路”就相当于IPv4中的子网），不能在不同子网中路由。结点使用链路本地地址与同一个链路上的相邻结点进行通信。例如，在没有路由器的单链路IPv6网络上，主机使用链路本地地址与该链路上的其他主机进行通信。

链路本地地址等效于169.254.0.0/16网段的自动专用IP寻址（APIPA）IPv4地址（在运行Windows系统的计算机上自动配置）。邻居发现过程要求使用链路本地地址，该地址始终自动配置（也就是无须手工配置），即使所有其他单播地址都不存在也是如此。图8-22所示为链路本地地址的结构。

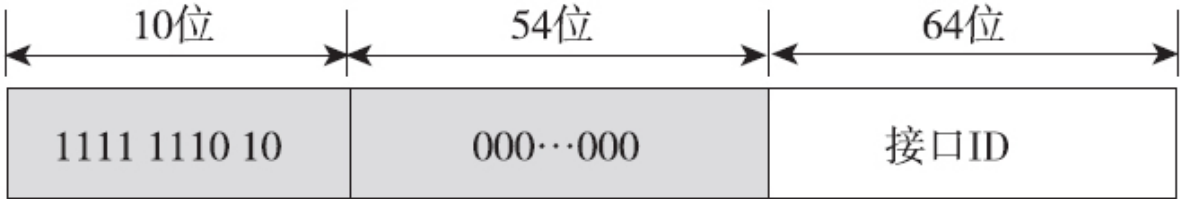


图 8-22 IPv6链路本地地址结构

从图中可以看出，链路本地地址始终以“1111 1110 10”（FE80）开头。后面紧跟着的54位均为0，最后的64位是用来标识接口，称为接口ID。在IPv6单播地址中的接口ID用来标识一个链路上的接口，不能在同一个链路上为不同结点分配相同的接口ID。所以对于链路本地地址的前缀始终是FE80::/64。IPv6路由器永远不会将链路本地通信转发出该链路。

(2) 站点本地地址

站点本地地址相当于IPv4中的局域网专用地址（如本章前面介绍的10.0.0.0/8、172.16.0.0/12、192.168.0.0/16这三个局域网专用IPv4地址段），仅可在本地局域网中使用。例如，没有与IPv6 Internet的直接路由连接的专用Intranet可以使用不会与全局地址冲突的站点本地地址。站点本地地址可以与全局单播地址配合使用，也就是在一个接口上可以同时配置站点本地地址和全局单播地址。但使用站点本地地址作为源或目的地址的数据报文不会被转发到本站点（相当于一个私有网络）外的其他站点。

站点本地地址的地址格式如图8-23所示，前48位总是固定的，即总是以FEC0::/48开始，因为站点本地地址的前10位固定为1111111011，后面紧接的是38位0。48个固定位之后，是16位可用于在组织中创建子网的子网ID。通过这16位地址空间，就可以在一个平滑的子网结构中最多拥有65 536个子网，或者细分子网ID字段的高位，以便创建分层的和可聚合的路由结构。子网ID字段之后，是64位的接口ID字段，用来标识子网上的特定接口，也是对应接口的MAC地址，与前面介绍的接口ID一样。

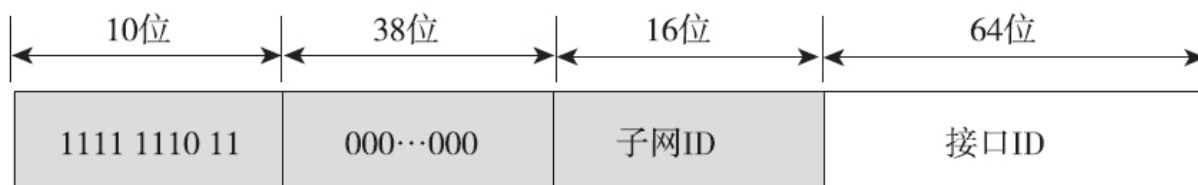


图 8-23 IPv6站点本地地址结构

经验之谈 站点本地地址是我们在局域网组建中经常要配置的IPv6地址，就像我们目前在IPv4网络中经常要配置局域网专用地址一样（其实站点就代表局域网），所以我们必须要记住站点本地地址的基本结构，它是以FEC0::/48为地址前缀的，后面的子网ID我们在设计局域网时就应设计好，为不同的子网分配连续或不连续的子网号；接口ID通常是通过复制对应接口的64位IPv6 MAC地址得到的。

从以上单播地址类型可以看出，IPv6具有分级地址模式，也就是在一个接口上可以同时配置应用于内部链路、站点（也就相当于内部网络）和Internet公网连接的至少两种单播地址。内部节点与Internet的连接不再需要经过IPv4时代的NAT技术进行地址转换，可直接为每台需要访问Internet的主机配置全局单播地址。

3.内嵌IPv4地址的IPv6单播地址

为了帮助从IPv4迁移到IPv6，促进两种类型主机的并存，定义了下列地址。

（1）IPv4兼容地址

在IPv6的转换机制中还包括了一种通过IPv4路由接口以隧道方式动态传递IPv6包的技术。这样的IPv6结点会被分配一个在低32位中带有全

球IPv4单播地址的IPv6全局单播地址。这种类型的地址就称为“与IPv4兼容和IPv6全局单播地址”。其格式如图8-24所示。注意这里的IPv4必须是全球唯一的公网IP地址。



图 8-24 兼容IPv4的IPv6全局单播地址格式

(2) IPv4映射IPv6本地单播地址

还有另一种嵌入IPv4的IPv6地址，用于局域网内部。这类地址用于把IPv4结点当作IPv6结点。这种类型的地址称为IPv4映射IPv6本地单播地址，格式如图8-25所示。

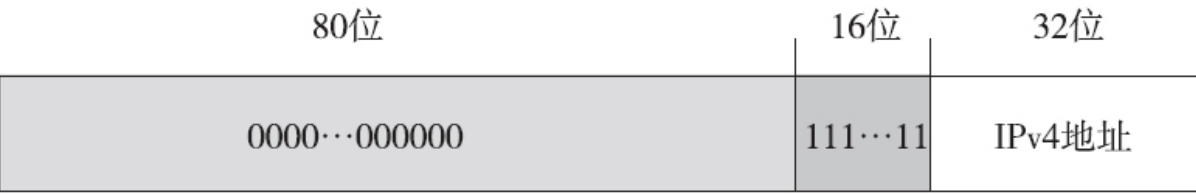


图 8-25 映射IPv4的IPv6本地单播地址格式

当处理拥有IPv4和IPv6结点的混合环境时，可以使用IPv6地址的另一种形式，即x:x:x:x:x:x:d.d.d.d，其中，“x”是IPv6地址的96位高位顺序字节的十六进制值，“d”是32位低位顺序字节的十进制值。而且前6个

16位地址块（96位）间是用冒号分隔的，后面的4个8位地址块（32）间仍用小圆点分隔。例如：0:0:0:0:0:0:10.1.2.3，或表示成::10.1.2.3。

（3）6to4地址

还有一种称为“6to4”的IPv6地址，用于在两个通过Internet同时运行IPv4和IPv6的结点之间进行通信。6to4地址由组合前缀2002::/16和该结点的32位公网IPv4地址构成，形成一个48位前缀。例如，IPv4地址131.107.0.1，6to4地址前缀是2002:836B:1::/48。当然，如果当前使用的是动态公网IPv4地址，则每次拨号接入Internet时得到的6to4 IPv6是不一样的。

4.两种特殊的单播地址

在IPv6单播地址中，还有两种特殊的单播地址：

（1）未指定地址

未指定地址（0:0:0:0:0:0:0:0或::）仅用于表示某个地址不存在。它等价于IPv4未指定地址0.0.0.0。未指定地址通常被用做尝试验证暂定地址唯一性数据包的源地址。未指定地址永远不会指派给某个接口或被用做目标地址。

（2）环回地址

环回地址（0:0:0:0:0:0:0:1或::1）用于标识环回接口，允许节点将数据包发送给自己。它等价于IPv4环回地址127.0.0.1。发送到环回地址的数据包永远不会发送给某个链接，也永远不会通过IPv6路由器转发。

8.5.2 IPv6组播地址

对于IPv6中的组播和任播地址，新的RFC3513标准中并没有针对原来的RFC2373标准进行太多改动。IPv6中的组播地址其实与IPv4中的组播地址是类似的。

IPv6组播地址可识别多个接口，对应于一组接口的地址（通常分属不同节点）。发送到组播地址的数据包被送到由该地址标识的每个接口。使用适当的组播路由拓扑，将向组播地址发送的数据包发送给该地址识别的所有接口。任意位置的IPv6节点可以侦听任意IPv6组播地址上的组播通信。IPv6节点可以同时侦听多个组播地址，也可以随时加入或离开组播组。

IPv6组播地址的最明显特征就是最高的8位固定为1111 1111。IPv6地址很容易区分组播地址，因为它总是以FF开始的。除了地址前缀外，组播地址还包括其他结构，以便标识它们的标志、作用域和组播组，如图8-26所示。

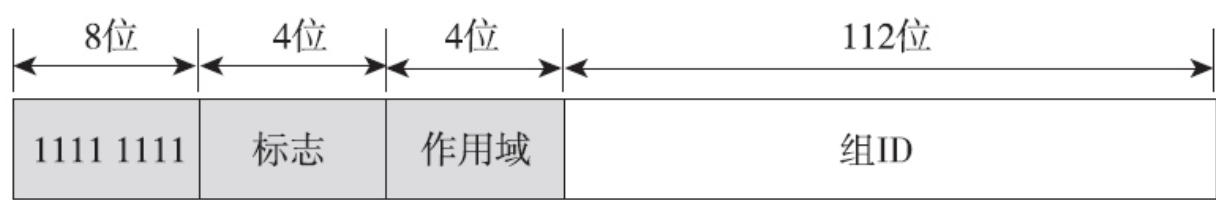


图 8-26 IPv6组播地址结构

组播地址结构中的各组成部分说明如下。

1) 标志：表示在组播地址上设置的标志，占4位。其实前3位均固定为0，只有最后1位具有标志意义，这一位记作“T”。当T设置为0时，表示该组播地址是由互联网号码指派机构IANA永久指派的、公认的组播地址；当T设置为1时，表示该组播地址是临时（非永久指派）组播地址。

2) 作用域：表示进行组播通信的IPv6网络的作用域，也就是本次组播通信的范围，占4位，路由器可以使用组播作用域来确定是否可以转发组播通信。在RFC 2373中定义的作用域如表8-4所示。

表 8-4 RFC2373 定义的 IPv6 组播地址作用域

字段值	作用域	字段值	作用域
0	保留	6、7	未分配
1	接口本地域	8	组织本地域
2	链路本地域	9~D	未分配
3	保留	E	全局域
4	管理本地域	F	保留
5	站点本地域		

例如，使用组播地址FF02::2的通信具有链路本地作用域，因为它的第四个十六进制位（作用域位）的值为2。IPv6路由器永远不会将此通信转发到本地链路以外。

3) 组ID：用于标识组播组，并且在作用域中是唯一的，占112位。永久指派的组ID独立于作用域，临时组ID仅与特定的作用域有

关。

为了标识接口本地作用域和链路本地作用域的所有节点，定义了下列组播地址：

□FF01::1（表示包括接口本地作用域所有节点的组播地址）；

□FF02::1（表示包括链路本地作用域所有节点的组播地址）。

为了标识接口本地作用域、链路本地作用域和站点本地作用域的所有路由器，定义了下列组播地址：

□FF01::2（表示包括接口本地作用域所有路由器的组播地址）；

□FF02::2（表示包括链路本地作用域所有路由器的组播地址）；

□FF05::2（表示包括站点本地作用域所有路由器的组播地址）。

因为在组ID中长达112位，所以可能有 2^{112} 个组ID。又因为IPv6组播地址在数据链路层封装时将被映射到以太网组播MAC地址，所以RFC 2373建议从IPv6组播地址的低阶32位指派组ID，并将剩余的原始组ID位设置为0，如图8-27所示。通过在组ID中只使用低位32位，每个组ID映射到唯一的以太网组播MAC地址。

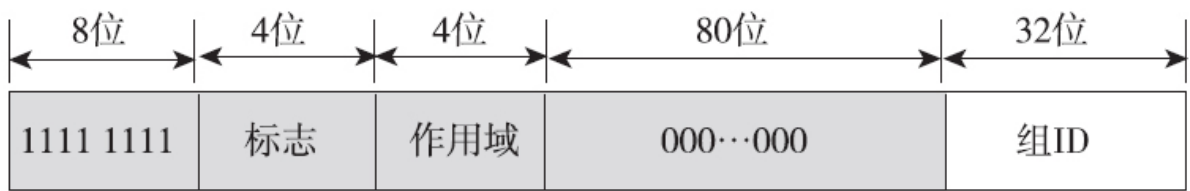


图 8-27 使用32位组ID的建议IPv6组播地址

8.5.3 IPv6任播地址

任播地址是IPv6协议中的全新概念，与上面的组播地址既有相似之处，又有本质区别。一个IPv6任播地址与组播地址一样也可以识别多个接口，对应一组接口的地址。大多数情况下，这些接口属于不同的节点。但是，与组播地址不同的是，发送到任播地址的数据包被送到由该地址标识的其中一个接口。由于使用任播地址的标准尚在不断完善中，所以目前很少有系统支持任意播送。

通过合适的路由拓扑，目的地址为任播地址的数据包将被发送到单个接口（该地址识别的最近接口，最近接口定义的根据是因为路由距离最近），而组播地址用于一对多通信，发送到多个接口。一个任播地址必须不能用作IPv6数据包的源地址；也不能分配给IPv6主机，仅可以分配给IPv6路由器。

目前，任播地址只作为目标地址使用，并且只分配给路由器。任播地址在单播地址空间之外分配，任播地址的作用域是分配的任播地址所属的单播地址类型的作用域。

子网-路由器任播地址已预定义，并且是必需的。该地址通过给定接口的子网前缀创建。若要构造子网-路由器任播地址，子网前缀中的位固定为相应的值，其他位则设置为0，如图8-28所示。任播地址中的

64位子网前缀是用来标识特定的链路。后面的64位是接口ID，语法与单播地址一样，只是把接口ID部分全部置0。

64位	64位
子网前缀	0000...0000

图 8-28 IPv6任播地址格式

发到子网-路由器任播地址的数据包将传送到子网中的一个路由器上。所有路由器的所有接口都要能支持该子网的任播地址。子网-路由器任播地址主要用于需要与任何一个路由器通信的应用节点，如一个网络中有几个网络出口，此时在网络中的应用服务器上配置好任播地址，就可以在任一时刻动态选择其中一个出口与外网连接。

8.5.4 IPv6主机和路由器地址

由于IPv6地址数过于庞大，加上IPv6地址类型又比较复杂，所以了解实际的网络配置中主机和网络设备配置的IPv6地址有哪些就显得非常重要了。安装单个网络适配器的IPv4主机，有一个指派给该适配器的单独的IPv4地址。但是，IPv6主机通常有多个IPv6地址，即使只有单一的接口。

IPv6主机通常在逻辑上是多主机的，因为它们至少有两个用于接收数据包的地址：一个为用于本地链接通信的链路本地地址，一个为可路由的站点本地地址或全局单播地址。

在IPv6主机上可以指派以下单播地址：

- ☐用于每个接口的链路本地地址；
- ☐用于每个接口的站点本地地址，或一个或多个全局单播地址；
- ☐用于回环接口的回环地址（::1）。

另外，每个主机都在以下组播地址侦听通信：

- ☐作用域为“节点本地”的所有节点组播地址（FF01::1）；

☐作用域为“链路本地”的所有节点组播地址（FF02::1）；

☐每个接口上用于每个单播地址请求的节点地址；

☐每个接口上已加入组的组播地址。

在IPv6路由器上可指派以下单播地址：

☐用于每个接口的链路本地地址；

☐用于每个接口的站点本地地址，或一个或多个全局单播地址；

☐用于回环接口的回环地址（::1）。

IPv6路由器可指派有以下任播地址：

☐用于每个子网的子网路由器任播地址；

☐其他任播地址（可选）。

另外，每个路由器将在以下组播地址侦听通信：

☐作用域为“节点本地”的所有节点组播地址（FF01::1）；

☐作用域为“节点本地”的所有路由器地址（FF01::2）；

☐作用域为“链路本地”的所有节点组播地址（FF02::1）；

☐作用域为“链路本地”的所有路由器组播地址（FF02::2）；

☐作用域为“站点本地”的所有路由器组播地址（FF05::2）；

☐每个接口上用于每个单播地址的请求节点地址；

☐每个接口上已加入组的组播地址。

从以上可以看出，站点本地地址和全局单播地址不能同时配置，如果主机或路由器接口要与Internet连接，就需要为它配置全局单播地址。

8.5.5 IPv6地址前缀表示形式

你可以像IPv4用地址前缀表示地址范围一样，IPv6也将地址范围表示为地址前缀。

IPv6地址前缀的基本表示形式为：地址前缀/前缀长度。地址前缀是指地址中最左边那些固定不变的部分采用原来的值，而可变部分置0得到，这与IPv4地址中的地址前缀获取方法是一样的。前缀长度就是地址前缀部分的位数。与IPv4地址前缀表示形式一样，在前缀符号“/”前一定要是一个完整的IPv6地址，否则会产生异义。

如要描述像12AB00000000CD3这样一个60位地址前缀，则可以有以下三种表示形式（不同IPv6地址可以表示的形式也不一样，这个地址的三种前缀表示形式其实就是一种，只是后两种采用了压缩零规则）：

□12AB:0000:0000:CD30:0000:0000:0000:0000/60;

□12AB::CD30:0:0:0:0/60;

□12AB:0:0:CD30::/60。

但是以下三种表示形式就不行了：

□12AB:0:0:CD3/60: 这是因为没有补齐整个地址的128位，前导0可以不写，但后面的0不能省略。

□12AB::CD30/60: 仅从这种格式来展开的话，这个地址就成了12AB:0000:0000:0000:0000:000:0000:CD30，而不是原来的地址格式了。

□12AB::CD3/60: 仅从这种格式来展开的话，这个地址就成了12AB:0000:0000:0000:0000:000:0000:0CD3，也不是前面的那个地址格式了。

在为IPv6主机、路由器配置IPv6地址时要注意前面列出的对应类型，同时要注意的是在一个接口上可以同时配置多个不同类型的IPv6地址，分别用于本地链路、本地站点和公网路由。但在配置这些IPv6地址时，要注意对应类型的地址前缀格式，不能配错。

8.6 IPv6地址自动配置

在本章前面就已提到，IPv6地址具有自动配置功能，所以虽然它长达128位，但配置起来并不是很复杂。IPv6最有用的方面是在不使用全状态配置协议（例如IPv6的动态主机配置协议DHCPv6）的情况下能够对自己进行自动配置。默认情况下，IPv6主机可以通过操作系统自动为每个接口配置的一个链路本地地址来进行本地寻址和邻居发现。通过使用路由器发现功能，主机也可以确定路由器的地址、附加地址和其他配置参数。

8.6.1 IPv6地址自动配置的类型

IPv6有以下三种自动配置类型，对于所有的自动配置类型，所配置的地址均是链接本地地址。

(1) 无状态的自动配置

这种IPv6地址的自动配置是基于路由器公告消息的接收。这些消息包括无状态地址前缀，并要求主机不使用全状态地址配置协议。

IPv6无状态地址配置协议是目前广泛采用的IPv6地址自动配置方式。配置了该协议的主机只需相邻路由器开启IPv6路由公告功能，即

可以根据公告报文包含的前缀信息自动配置本机地址。但无状态地址配置方案中路由器并不记录所连接的IPv6主机的具体地址信息，可管理性差。而且当前无状态地址配置方式不能使IPv6主机获取DNS服务器的地址和域名等配置信息，在可用性上有一定缺陷。对于互联网服务提供商（ISP）来说，也没有相关的规范指明如何向路由器自动分配IPv6前缀，所以在部署IPv6网络时，只能采用手动配置的方法为路由交换设备配置IPv6地址。

（2）监控状态的自动配置

监控状态的自动配置是基于全状态地址配置协议的使用（例如DHCPv6）来获取地址和其他配置选项的。当主机收到不包括地址前缀的路由器公告消息，并要求主机使用全状态地址配置协议时，将使用全状态地址配置。当本地链路上没有路由器存在时，主机也使用全状态地址配置协议。

DHCPv6是动态主机配置协议（DHCP）的IPv6版本，协议基本规范由RFC3315定义。相对于IPv6无状态地址自动配置协议，DHCPv6属于一种有状态地址自动配置协议。在有状态地址配置过程中，DHCPv6服务器分配一个完整的IPv6地址给主机，并提供DNS服务器地址和域名等其他配置信息，这中间可能通过中继代理转交DHCPv6报文，而且最终服务器能把分配的IPv6地址和客户端的绑定关系记录在案，从而增强了网络的可管理性。

DHCPv6服务器也能提供无状态DHCPv6服务，即DHCPv6服务器不分配IPv6地址，仅需向主机提供DNS服务器地址和域名等其他配置信息，主机IPv6地址仍然通过路由器公告方式自动生成，这样配合使用就弥补了IPv6无状态地址自动配置的缺陷。DHCPv6协议还提供了DHCPv6前缀代理的扩展功能，上游路由器可以自动为下游路由器分派地址前缀，从而实现了层次化网络环境中IPv6地址的自动规划，解决互联网提供商（ISP）的IPv6网络部署问题。

（3）两者兼备的自动配置

这种自动配置类型是根据路由器公告消息的回执进行配置。这些消息包括无状态地址前缀，并要求主机使用全状态地址配置协议。

8.6.2 自动配置过程

IPv6节点的链路本地地址自动配置过程如下：

- 1) 基于链路本地前缀FE80::/64和64位接口标识组合成暂定的链路本地地址。
- 2) 执行重复地址检测，以便验证暂定链路本地地址的唯一性。
- 3) 如果重复地址检测失败（也就是链路中有与本暂定链路本地地址一样的地址），则必须对该节点执行手动配置。
- 4) 如果重复地址检测成功（也就是链路中没有与本暂定链路本地地址一样的地址），则此暂定链路本地地址将是唯一的和有效的，并为该接口初始化链路本地地址。

对于IPv6主机地址自动配置，在节点完成自动配置过程后，还需继续如下过程：

- 1) 主机发送路由器请求消息。
- 2) 如果没有收到路由器公告消息，则主机将使用全状态地址配置协议来获取地址和其他配置参数。

3) 如果收到路由器公告消息，将在主机设置消息中包括配置信息。

4) 对于每个无状态自动配置，使用以下方法来验证自动配置的地址：

□使用地址前缀和适当的64位接口标识派生出的暂定地址。

□使用重复地址检测来验证暂定地址的唯一性。

如果正在使用该暂定地址，则不能为该接口初始化此地址；如果没有使用该暂定地址，则初始化该地址，包括基于路由器公告消息中包括的信息来设置有效状态和首选状态的生存时间；如果在路由器公告消息中指定，则主机将使用全状态地址配置协议来获取附加地址或配置参数。

图8-29所示的是一台Windows 7主机自动获得的四个IPv6地址。其中第一个以2002开头的IP地址是一个6to4 IPv6地址；第二个是本地站点IPv6地址，相当于局域网IPv6地址；第三个是临时IPv6地址，即动态拨号接入时从ISP中获得的临时IPv6地址，可以看出它也是一种6to4地址；第四个是本地链接IPv6地址，仅于本地链路通信，后面的“%”跟的是逻辑接口号。

```
命令提示符

PPP 适配器 Unet_PPPoE:

连接特定的 DNS 后缀 . . . . . :
IPv4 地址 . . . . . : 113.240.9.47
子网掩码 . . . . . : 255.255.255.255
默认网关 . . . . . : 0.0.0.0

以太网适配器 本地连接:

连接特定的 DNS 后缀 . . . . . :
IPv6 地址 . . . . . : 2002:71f0:92f:b:9d0d:eafc:823b:3905
本地站点的 IPv6 地址 . . . . . : fec0::b:9d0d:eafc:823b:3905%2
临时 IPv6 地址 . . . . . : 2002:71f0:92f:b:3533:2e87:5f49:77d6
本地链接 IPv6 地址 . . . . . : fe80::9d0d:eafc:823b:3905%11
IPv4 地址 . . . . . : 192.168.1.10
子网掩码 . . . . . : 255.255.255.0
默认网关 . . . . . :

以太网适配器 VMware Network Adapter VMnet1:

连接特定的 DNS 后缀 . . . . . :
本地链接 IPv6 地址 . . . . . : fe80::4599:8338:d494:372b%13
IPv4 地址 . . . . . : 192.168.112.1
子网掩码 . . . . . : 255.255.255.0
```

图 8-29 一个节点上获得的四个IPv6地址

第9章 路由协议及工作原理

本章内容相对较为复杂，对于初学者来说，可暂时作为选学内容。

在第7章我们已介绍了网络层的主要功能和主要路由算法，本章将介绍目前常见的几种动态路由协议（包括RIP、OSPF、IS-IS和BGP）的基础知识、所采用的路由算法工作原理、主要路由消息及报文格式。其中最重要的是使用这些路由协议的基本网络结构、路由表基本生成原理，以及不同路由消息报文的用途和基本报文格式。

为了把复杂的内容讲解得尽可能通俗易懂，本章仍采用前面各章的通俗介绍方法，里面不仅有大量的比喻，还包括了大量的示例、图表。

9.1 RIP路由协议

RIP（Routing Information Protocol，路由信息协议）是应用较早、使用较普遍的内部网关协议（Interior Gateway Protocol，IGP），适用于小型同类网络的一个自治系统（AS）内的路由信息的传递。RIP协议采用距离矢量（也就是用距离作为路由变量）路由算法，使用跳数（即Hop）来衡量到达目标地址的路由距离，且最大跳数值仅为15。

RIP协议现在有两个版本，即RIPv1和RIPv2，新版本RIPv2支持纯文本和MD5身份认证、路由汇总、无类别域间路由（Classless Interdomain Routing，CIDR）和可变长度子网掩码（Variable-Length Subnet Masks，VLSM）。有关CIDR和VLSM请参见本书第8章。

9.1.1 RIP路由度量机制

每一种路由协议都想通过某一个参考变量来实现最佳路由的计算，也就是每一种路由协议都有自己的路由度量机制。不同的路由协议所采取的路由度量机制是不一样的，这取决于各自所采用的路由算法。

1.理解“跳数”

RIP协议采用的是“距离矢量”路由算法，是仅以“距离”作为度量标准的，也就是路由的“距离”是RIP路由的度量。但要注意的是，这里“距离”不是指实际的通信线路长度，也不是两个网络或者两台主机相距的物理距离，而是指除源网络所连接的路由器外，到达目的网络的整条路由路径中所经过的路由器数，即跳数（Hop Count）。

假设在图9-1所示的网络中，四个路由器都运行了RIP协议。现在要配置从PC1所在网络到达PC2所在网络的RIP路由，则跳数就是3（R1不算，其他每个路由器算一跳）。但是我们看一下，这四个路由器所

连接的网络数达到了5个（从1.1.1.0/24到5.1.1.0/24），当然这还只是最简单的网络了，因为每个路由器还可以连接多个网络。但在同一个路由器上直接连接的多个网络彼此间的度量值为0，因为它们是直连路由。

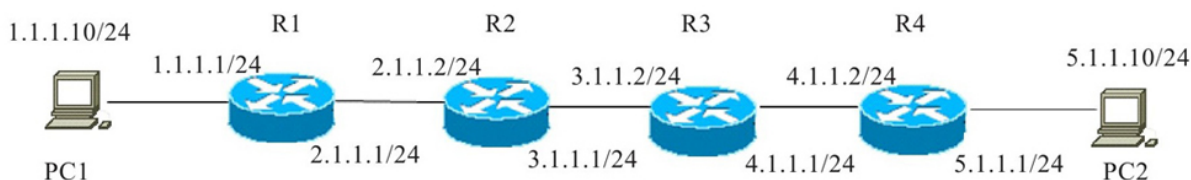


图 9-1 RIP路由跳数计算示例

2.度量更新规则

RIP路由器采用完整路由表更新方式，也就是每个RIP路由器会把自己的路由表发给相邻的RIP路由器，以此来进行彼此的路由表更新。总体来说，它遵循以下几个基本原则：

□路由表项每经过一次邻居之间的传递，其度量值加1（最大值为15，下同）。

在如图9-1所示的网络中，R1路由器把它所连接的1.1.1.0/24网络向R2通告，这时在R2上到达1.1.1.0/24网络的度量为1，当R2再把它获取的到达1.1.1.0/24网络的路由发给R3时，R3上到达1.1.1.0/24网络的度量就变为2，而当R3把获取的到达1.1.1.0/24网络的路由发给R4时，R4上到达1.1.1.0/24网络的度量就变为3。

□收到新路由表项时，在路由表中添加新的路由表项，其度量是在接收的路由表项度量基础上加1，同时在新添加的路由表项中标注其下一跳地址就是发送路由更新的邻居路由器的接口。

如图9-1所示的示例中，假设R4在收到来自R3的路由表更新时发现包含了一条到达6.1.1.0/24网络的新路由表项，并且其度量为2，这时R3就会在自己的路由表中添加这条新的路由表项，并且设置度量为3（2+1），“下一跳地址”为R3与R4相连的那个接口IP地址。

□收到原有路由表项的路由更新时，先对有更新的路由表项的度量加1，然后与对应的路由表项中原度量进行比较，仅接收度量值更小或相等的更新，忽略度量值比原来的值更大的路由更新。

假设在图9-1中，R2中原有的路由表项如图9-2所示，现又收到来自R3的如图9-3所示的路由更新（启用了水平分割功能后就不会出现这样的更新了）。

目的网络	下一跳	距离
2.1.1.0	--	0
3.1.1.0	--	0
4.1.1.0	3.1.1.2	1
5.1.1.0	3.1.1.2	2

图 9-2 R2路由器上原来的路由表

目的网络	下一跳	距离
2.1.1.0	3.1.1.1	1
3.1.1.0	--	0
4.1.1.0	--	0
5.1.1.0	4.1.1.2	1

图 9-3 R2收到来自R3的路由更新

对于R2来说，在收到来自R3的路由更新后，首先对图9-3所示的各路由表项的度量加1，结果得到的路由表项如图9-4所示。对比图9-4和图9-2可以看出，到达2.1.1.0网络和3.1.1.0网络的度量值比原来的还大，所以忽略更新，而到达4.1.1.0网络和5.1.1.0网络的度量值是相等的，进行路由表项更新，所以最终R2上的路由表还是图9-2所示那样。

目的网络	下一跳	距离
2.1.1.0	3.1.1.2	2
3.1.1.0	3.1.1.2	1
4.1.1.0	3.1.1.2	1
5.1.1.0	3.1.1.2	2

图 9-4 R2对收到的路由更新表项度量值加1后的路由表

如果一个接口连接的网络没有指定（也就是没有宣告所直接连接的网络），则它不会在任何RIP更新中被通告。如图9-1中，在R1的配

置中没有宣告它所连接的1.1.1.0/24网络，则其他路由器上也就没有到达这个网络的RIP路由表项，如果这些路由器仅启用了RIP协议，且不配置静态路由，则其他路由器也就不能到达1.1.1.0/24这个网络了。

当然，到达同一目的网络，也可能距离是一样的，这时就可能存在同一目的网络的两条RIP路由表项，可以实现负载均衡。

9.1.2 RIP路由更新机制

RIP协议有两种更新机制：一是定期更新，二是触发更新。定期更新是根据设置的更新计时器定期发送**RIP**路由通告。该通告报文中携带了除水平分割机制抑制的**RIP**路由之外的本地路由器中所有的**RIP**路由信息。而触发更新则是**RIP**路由器仅在路由表项发生变化时发送的**RIP**路由通告，该通告报文中仅携带本地路由表中有变化的路由信息。**RIP**路由器一旦察觉到网络变化，就尽快甚至是立即发送更新报文，而不等待更新周期结束。只要触发更新的速度足够快，就可以最大程度地防止计数到无穷大的情况发生，但是这一现象还是有可能发生的。

无论是定期更新还是触发更新，**RIP**路由更新都有如下规则：

□如果更新的某路由表项在路由表中没有，则直接在路由表中添加该路由表项；

□如果路由表中已有相同目的网络的路由表项，且来源端口相同，那么无条件根据最新的路由信息更新其路由表；

□如果路由表中已有相同目的网络的路由表项，但来源端口不同，则比较它们的度量值，将度量值较小的一个作为自己的路由表项；

□如果路由表中已有相同目的网络的路由表项，且度量值相等，则保留原来的路由表项。

下面主要介绍RIP路由的定期更新机制。

1.RIP路由定期更新机制

RIP路由器总是会每隔30s（这是默认值，可以修改，而且也可能与设置值有些偏差）通过UDP 520端口以RIP广播应答方式向邻居路由器发送一个路由更新包，包中包括了本路由器上的完整的路由表（除了被水平分割机制抑制的路由表项），用来向邻居路由器提供路由更新，同时用来向邻居路由器证明自己的存在。RIP的路由表中主要包括目的网络、下一跳地址和距离这三个字段，如图9-4所示。

如果一个路由器在180s（这也是默认值，可以修改）内没有收到某个邻居路由器发来的路由更新，则这个路由器就会标记该邻居路由器为不可达路由器，使这个邻居路由器处于抑制周期。当路由器处于抑制周期内，它仍然用于向前转发数据包，但网络中的其他路由器不学习到达该路由器所连网络的路由信息，除非是一条更好的到达该路由器所连网络的路由信息，如本来是3跳，在抑制周期内学到了一条2跳的路由信息。但抑制周期过后，即使是差的路由信息也会被接受。

如果在连续的240s（这也是默认值，可以修改）内还没收到这个路由器的路由更新，则本地路由器会在路由表中删除与该邻居路由器

相关的路由表项。

由此可见，这个路由更新不仅影响着整个RIP网络中的路由器上路由表的更新和所有需要到达或者经过该路由器的数据包路由，还影响着其他邻居路由器是否当它存在。试想一下，如果有一个数据包是要发送到连接某个RIP路由器的网络的一台主机上，但这台RIP路由器当时恰好出现了故障，没有这个路由器更新机制的话，其他路由器也就不知道它当前出现了故障，仍按原来的路由路径传输数据包，结果当然是数据包总是无法到达目的主机了，尽管可能经过多次尝试。

2.RIP路由更新机制解析示例

为了更好地理解RIP协议路由表的更新机制，下面以图9-5所示的简单的互连网络为例来讨论路由表是怎样建立的。

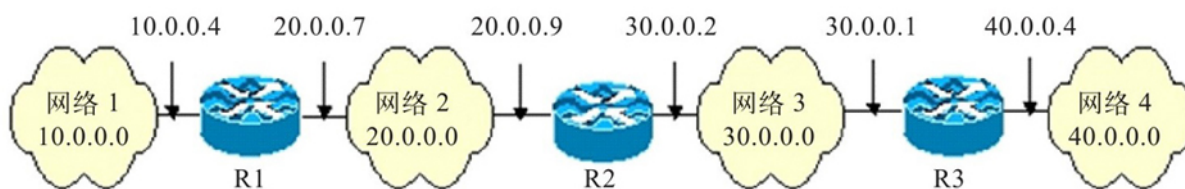


图 9-5 RIP路由表建立网络示例

1) 在一开始，所有路由器中的路由表只有自己所直接连接的网络的路由表项信息。但不是RIP路由表项，是直连路由表项，无需下一跳（用“--”表示），度量“距离”也均为0，各路由器的初始路由表如图9-6所示，均只有两条直连网络的路由表项。

R1 的路由表			R2 的路由表			R3 的路由表		
目的网络	下一跳	距离	目的网络	下一跳	距离	目的网络	下一跳	距离
10.0.0.0	--	0	20.0.0.0	--	0	30.0.0.0	--	0
20.0.0.0	--	0	30.0.0.0	--	0	40.0.0.0	--	0

图 9-6 R1、R2和R3的初始路由表

2) 接下来，各路由器就会按设置的周期（默认为30s）向邻居路由器发送路由更新了。具体哪个路由器会先发送路由更新，取决于路由器启动顺序。现假设路由器R2先收到来自路由器R1和R3的路由更新，然后其会更新自己的路由表，如图9-7所示。从中可以看出，它新添加了分别通过R1和R3到达10.0.0.0网络和30.0.0.0网络的路由表项，度量值均为1，因为它只经过了一跳。

目的网络	下一跳	距离
20.0.0.0	--	0
30.0.0.0	--	0
10.0.0.0	20.0.0.7	1
40.0.0.0	30.1.1.1	1

图 9-7 R2路由更新后的路由表

3) R2更新自己的路由表后，会把完整的路由表发给邻居路由器R1和R3。路由器R1和R3分别再进行更新。根据前面介绍的RIP路由表更新的规则可以知道，R1首先是把从R2上接收到的路由表的相应度量项加1，得到的路由表如图9-8所示。

目的网络	下一跳	距离
20.0.0.0	20.0.0.9	1
30.0.0.0	20.0.0.9	1
10.0.0.0	20.0.0.9	2
40.0.0.0	20.0.0.9	2

图 9-8 R1对收到的来自R2路由表进行度量加1后形成的路由表

4) 然后R1再把图9-8所示的路由表与自己原来的路由表（图9-6中的上图所示）进行比较，凡是新添加的，和度量值小于等于原来的路由表项均更新，度量值更大的路由表项将忽略更新。经过比较发现有两条新的路由表项，其目的网络分别为30.0.0.0和40.0.0.0，直接在路由表中添加。而原来已有的两条10.0.0.0和20.0.0.0表项，发现路由度量值1比原来的0还大，忽略更新，结果就得到R1更新后的路由表，如图9-9所示。

目的网络	下一跳	距离
10.0.0.0	--	0
20.0.0.0	--	0
30.0.0.0	20.0.0.9	1
40.0.0.0	20.0.0.9	2

图 9-9 R1在收到R2路由更新后的路由表

用同样的方法，可以得出R3在收到R2路由更新后的路由表如图9-10所示。

目的网络	下一跳	距离
30.0.0.0	--	0
40.0.0.0	--	0
10.0.0.0	30.0.0.2	2
20.0.0.0	30.0.0.2	1

图 9-10 R3在收到R2路由更新后的路由表

但RIP路由协议存在一个问题，那就是网络收敛比较慢，当网络出现故障时，要经过比较长的时间才能将此信息传送到所有的路由器，而且中间有许多是无效路由更新。

仍以图9-5为例，现在三个路由器都已经建立了各自的稳定路由表，假设R1路由器和网1（10.0.0.0）的连接线路断开了。此时R1可以立即发现，并更新自己的路由表，将到10.0.0.0的路由表项距离改为16（即不可达），并在30s后将此路由更新信息发给R2。但是，R2从R3得到的路由更新是“经过R2到达10.0.0.0网络的距离为2”，明显度量值更小，于是R2将此路由表项更新为“经过R3到达10.0.0.0的距离为3”，然后再通过路由更新发给R3，此时R3的路由表中更新为“经过R2到达10.0.0.0网络的距离为4”。R3再通过路由更新发给R2信息，结果是“经

过R3到达10.0.0.0网络的距离为5”，一直如此反复，直到该路由表项的距离达到16，R2和R3才知道10.0.0.0网络是不可达的。

9.1.3 RIP路由收敛机制

任何距离向量类路由选择协议（**RIP**也是这类路由协议）都有一个问题，路由器不知道网络的全局情况，路由器必须依靠相邻路由器来获取网络的可达信息。由于路由更新信息在网络上传播慢，所以所有距离矢量路由算法都有一个收敛慢的问题，这个问题将导致网络中各路由器路由信息不一致的现象产生。**RIP**协议使用以下机制可以减少因网络上的不一致性带来的路由选择环路的可能性。

1.记数到无穷大机制

RIP协议允许最大跳数值为15。大于15的目的地被认为是不可达的。这个数字在限制了网络大小的同时也防止了一个称为“记数到无穷大”的问题。记数到无穷大机制的工作原理如图9-11所示。

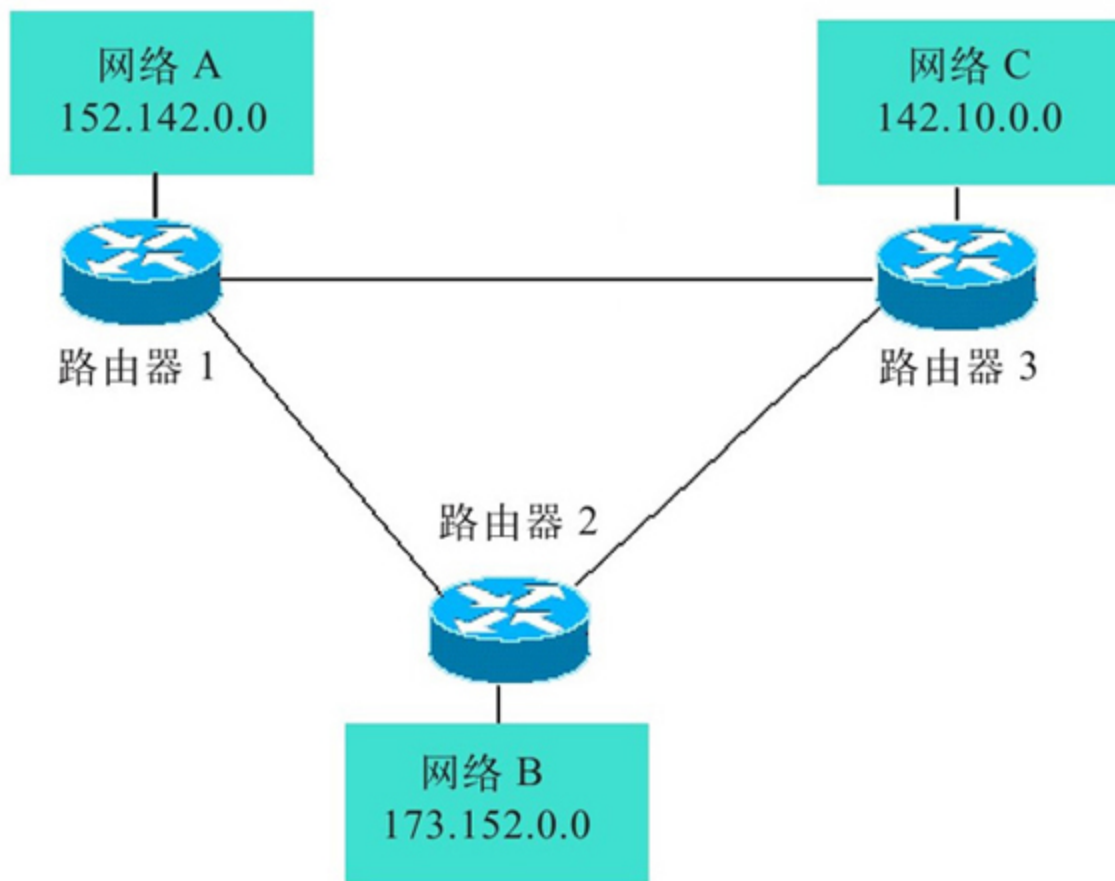


图 9-11 路由器收敛机制示例

1) 现假设路由器1断开了与网络A的连接，此时路由器1会立即产生一个路由更新向其邻居路由器2和路由器3通告，告诉它们，路由器1不再有到达网络A的路径。假设这个更新信息传输到路由器2被推迟了（CPU忙、链路拥塞等），但到达了路由器3，路由器3会立即从路由表中去掉到网络A的路径。

2) 但由于路由器2没有收到路由器1的这个路由更新信息，于是它仍会定期向它的邻居（包括路由器1和路由器3）发送路由更新信息，

通告网络A是以2跳的距离可达。路由器3收到这个更新信息后，认为出现了一条通过路由器2到达网络A的新路径，于是路由器3告诉路由器1，它能以3跳的距离到达网络A。

3) 路由器1在收到路由器3的路由更新后，就把这个信息加上一跳后向路由器2和路由器3同时发出更新信息，告诉它们路由器1可以以3跳的距离到达网络A。

4) 此时路由器2在收到路由器1的消息后，比较发现与原来到达网络A的路径不符，更新成可以以4跳的距离到达网络A。这个消息再次会发往路由器3，以此循环，直到跳数达到超过RIP协议允许的最大值（在RIP中定义为15）。一旦一个路由器达到这个值，它将声明这条路径不可用，并从路由表中删除此路径。

由于记数到无穷大问题，路由选择信息将从一个路由器传到另一个路由器，每次跳数加1。路由选择环路问题将无限制地进行下去，除非达到某个限制。这个限制就是RIP的最大跳数。当路径的跳数超过15，这条路径才从路由表中删除。

2.水平分割法

水平分割就是使路由器不向对应路由更新表项输入的方向回传此条路由表信息，使它只沿一个方向通告。通俗地讲就是，如果一条路

由信息是从某个端口学习到的，那么从该端口发出的路由更新中将不再包含该条路由信息，其目的就是为了避免出现路由更新环路。

在图9-12所示的网络中，路由器2通过帧中继连接路由器1和路由器3，两个PVC（永久虚电路）都在路由器2的同一个物理接口（S0）中止。如果在路由器2启用了水平分割功能，那么路由器3将收不到路由器1的路由选择信息，反之亦然。因为路由器1，或路由器3向路由器2通告的路由更新信息不会再从S0接口向外通告给路由器3或路由器1。

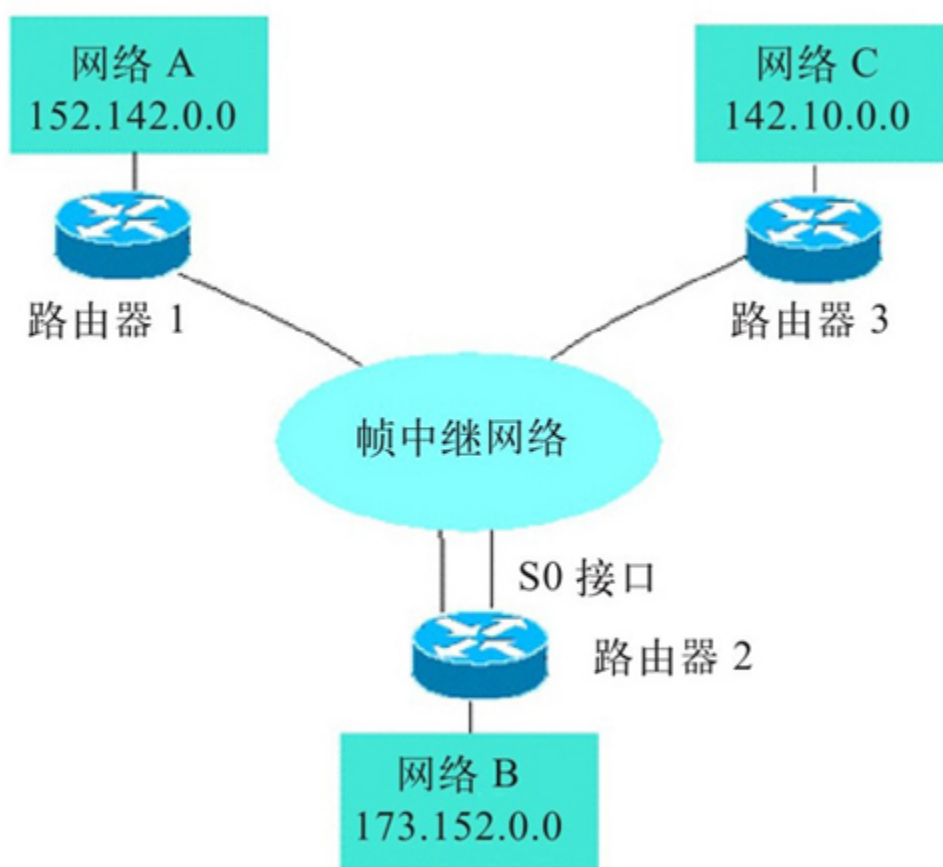


图 9-12 水平分割示例

在如图9-5所示的网络中，如果R2启用了水平分割功能，则R2在收到R1发来的包括10.0.0.0网络的路由更新后，在向R1发送的路由更新中就不会再包括10.0.0.0对应的路由表项了，这样就可以避免路由表发送的死循环出现。

3.毒性逆转水平分割法

水平分割功能是路由器用来防止把一个接口得来的路径又从此接口传回，导致路由更新环路的出现。毒性逆转水平分割方法是在更新信息中包括这些回传路径，但会把这些回传路径的跳数设为16（无穷）。通俗地讲就是，如果一条路由信息是从某个端口学习到的，那么从该端口发出的路由更新分组中将继续包含该条路由信息，但将这条信息的metric置为16。通过把跳数设为无穷，并把这条路径告诉源路由器，能够更快地消除路由信息的环路。但它增加了路由更新的负担。

4.保持定时器法

保持定时器法是设置路由信息被抑制的时间，默认为180s。当路由器接收到一个不可达的路由更新时，路由器将会把这条路由更新置于无效抑制状态，不再接收对应路由的更新信息，也不向外发送这条

路由更新信息，一直持续到接收到一个带有更好度量的对应路由更新分组或者相应保持计时器到期为止。

在图9-12所示的网络中，由于线路原因，从路由器1发往路由器2的路由更新被延迟到达，致使路由器2不能及时更新，所以路由器2仍会将新的错误路由信息发送给路由器3。但使用了保持计数器法后，这种情况将不会发生，因为路由器3在收到来自路由器1的网络A不可达的路由更新后，将在180s内不接受通向网络A的新的路由信息，而经过这段时间后，路由器2也已正确进行了更新，将不会再发送错误的路由信息给路由器3。

9.1.4 RIP报文格式

前面说了，RIP使用UDP报文在邻居路由器间交换路由表，所使用的UDP端口号为520。通常情况下RIPv1报文为广播报文；而RIPv2报文为组播报文，组播地址为224.0.0.9，默认每隔30s向邻居路由器发送一次路由更新报文。如果设备经过180s没有收到来自对端的路由更新报文则将所有来自此设备的路由信息标志为不可达，若在240s内仍未收到更新报文就将这些路由从路由表中删除。如图9-13所示为RIP路由更新报文格式，注意其中v1版本和v2版本的不同。

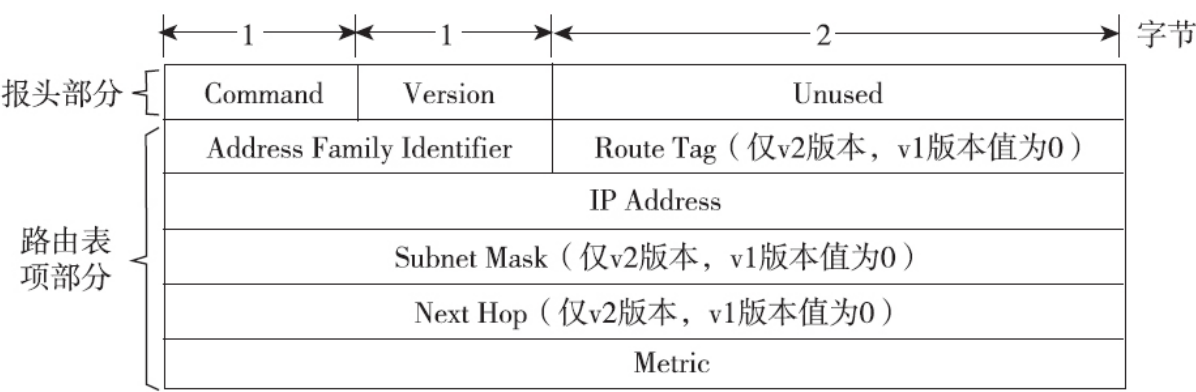


图 9-13 RIP协议报文格式

(1) Command

命令字段，占1字节，用来指定数据报分组是请求分组还是响应分组。在v1版本中主要有以下四种分组类型：Request（请求，对应值为1）、Response（响应，对应值为2）、Traceon（启用跟踪标记，对应

值为3，自v2版本后已经淘汰）、Traceoff（关闭跟踪标记，对应值为4，自v2版本后已经淘汰）。请求分组是请求邻居路由器发送全部或部分路由表信息的分组，响应分组可以是路由器主动提供的周期性路由更新分组或者对请求分组的响应。

（2）Version

版本字段，占1字节，v1版本值为1，v2版本值为2。

（3）Unused

未使用的字段，占2字节，值固定为0。

（4）Address Family Identifier（AFI）

地址族标识符字段，占2字节，指出所使用的地址族。RIP设计用于携带多种不同协议的路由信息，每个项都有地址族标志来表明使用的地址类型，IP地址的AFI是2。

（5）Route Tag

路由标记字段，占2字节，提供区分内部路由（由RIP学得）和外部路由（由其他协议学得）的方法。它携带着一个EGP和BGP的自治系统号。该字段仅在v2版本使用，使用对应外部路由的标记值，v1版本

不用（值固定为0），因为v1版本不支持由其他类型路由重新发布为RIP路由。

（6）IP Address

IP地址字段，占4字节，指定路由的目的网络地址，可以是标准网段地址、子网地址。

（7）Subnet Mask

子网掩码字段，占4字节，指定目的网络的子网掩码，仅在v2版本中使用，v1版本中该字段值固定为0。因为v1版本不支持无类别网络，也就是不支持子网，仅支持标准的有类网络，此时具体网段的子网掩码是固定的。

（8）Next Hop

下一跳字段，占4字节，指出RIP路由的下一跳的IP地址。如果为0.0.0.0，则表示发布此条路由信息的路由器地址就是最优下一跳地址。该字段也仅在v2版本中使用，v1版本中该字段值固定为0，因为v1版本采用的是广播方式发送，无具体的下一跳。

（9）Metric

RIP路由的Metric（度量）值字段，也就是“距离”值，占4字节，最大有效值为15，16时表示该路由不可达了。

说明 在以上整个RIP报文各字段中可分为头部（Header）和路由表项（Route Entries）两大部分。头部包括Command、Version和Unused这三个字段，其余字段都属于路由表项。在一个RIP报文中，最多可以有25个路由表项，也就是在一个RIP分组中最多可含有25个地址项，即一个分组中最多可一次性通告25条RIP路由表项。

9.2 OSPF路由协议

由于RIP协议主要用于小型网络，在20世纪80年代中期就已不能适应大规模异构网络的互连，一种互连功能更强大的路由协议——OSPF（Open Shortest Path First，开放式最短路径优先协议）就随之产生了。它是Internet工程任务组织（IETF）的内部网关协议工作组为IP网络而开发的一种路由协议，作为RIP的后继内部网关协议，目前它的最新版本是第4版，即OSPFv4。

9.2.1 OSPF协议简介

OSPF也是一个内部网关协议，用于单个自治体系（AS）的路由器之间。OSPF采用链路状态技术，路由器互相发送直接相连的链路信息和它所拥有的到其他路由器的链路信息。每个OSPF路由器维护相同AS拓扑结构的数据库。从这个数据库里，构造出最短路径树来计算出路由表。当拓扑结构发生变化时，OSPF协议能迅速重新计算出路径，而只产生少量的路由协议流量。此外，所有的OSPF路由选择分组的交换都是经过验证的。

OSPF协议与RIP协议相比要复杂许多，主要表现在以下几个方面：

(1) 采用链路状态路由算法

RIP采用相对简单的“距离矢量”路由算法，仅以“距离”作为度量；而OSPF采用的是更加复杂的“链路状态”路由算法，考虑的因素非常多，包括线路带宽、端口状态、端口开销、端口优先级、链路稳定性等。

(2) 可划分不同区域

虽然在一个OSPF网络中，只能有一个AS，但在这个AS中可以划分多个区域（Area），同一个区域的不同OSPF路由器进程可以一样，也可以不一样。在OSPF路由协议中，每一个区域中的路由器都按照该区域中定义的链路状态算法来计算网络拓扑结构，这意味着每一个区域都有着该区域独立的网络拓扑数据库及网络拓扑图。对于每一个区域，其网络拓扑结构在区域外是不可见的，同样，在每一个区域中的路由器对其域外的其余网络结构也不了解。这样做有利于减少网络中链路状态数据包在全网范围内的广播，也是OSPF将其路由域或一个AS划分成很多个区域的重要原因。

(3) 有不同路由器角色

正因为OSPF可以划分为多个不同的区域，所以就涉及相同区域内部，以及不同区域间的路由问题。这也就决定了在OSPF网络中存在多

种不同类型的路由器，并担当不同角色，如区域内部路由器、区域边界路由器、AS边界路由器等。具体将在本章后面专门介绍。

（4）收敛性能更高

在RIP协议中，所有的路由都由跳数来描述，到达目的地的路由最大不超过15跳，这就限制了RIP的服务半径，即其只适用于小型的简单网络。同时，运行RIP协议的路由器需要定期（默认为30s）在网络邻居路由器上通告自己的整个路由表信息，以便及时对网络拓扑结构的改变进行收敛。这样的收敛方式，不仅速度慢，而且极容易引起通告风暴（尤其是在路由更新包发送频率加大时）或者造成路由环路等问题。

OSPF是基于链路状态的路由协议，它克服了RIP路由收敛慢的缺陷。因为OSPF路由器不再交换整个路由表，而是同步各路由器对网络状态的认识，即链路状态数据库，然后通过Dijkstra最短路径算法计算出网络中各目的地址的最优路由。这样OSPF路由器间不需要定期交换大量数据，而只保持着一种连接，仅在链路状态发生变化时，才通过组播方式对这一变化做出反应。这样不但减轻了系统的负荷，而且达到了对网络拓扑的快速收敛，因此OSPF路由协议即使是在大型网络中也能够较快地收敛。而这些正是OSPF强大生命力和应用潜力的根本所在。

总体来说，OSPF具有如下特点：

□适应范围广——支持各种规模的网络，最多可支持几百台路由器。

□快速收敛——在网络的拓扑结构发生变化后立即发送更新报文，使这一变化在自治系统中同步。

□无自环——由于OSPF根据收集到的链路状态用最短路径树算法计算路由，从算法本身保证了不会生成自环路由。

□支持区域划分——允许自治系统的网络被划分成区域来管理，区域间传送的路由信息被进一步抽象，从而减少了占用的网络带宽。

□支持等价路由——支持到同一目的地址的多条等价路由。

□支持路由分级划分——使用4类不同的路由，按优先顺序来说依次是区域内路由、区域间路由、第一类外部路由、第二类外部路由。

□支持验证——支持基于接口的报文验证，以保证报文交互和路由计算的安全性。

□支持组播发送——在某些类型的链路上可以以组播地址形式发送协议报文，减少对其他设备的干扰。

9.2.2 OSPF的AS与Area

学习OSPF路由，首先要从AS（自治系统）和Area（区域）这两个概念学起，否则你根本无法理解本章后面将要介绍的各种OSPF技术和路由工作原理。

1.理解AS

AS是一组使用相同路由协议交换路由信息的路由器，又称路由域（Routing Domain）。它可以是一个路由器直接连接到一个LAN上，然后连到Internet上；也可以是一个由企业骨干网互连的多个局域网。同一个AS中的所有路由器必须相互连接，运行相同的路由协议，同时分配同一个AS号。

在OSPF网络中，只有在同一AS中的路由器才会相互交换链路状态信息；在同一个AS中，所有的OSPF路由器都维护一个相同AS结构描述的数据库。该数据库中存放的是路由域中相应链路的状态信息。OSPF路由器正是通过这个数据库计算出其OSPF路由表的。而且OSPF可以将一个AS分割成多个小的区域，便于网络的拓展，而且每个OSPF路由器只在区域内部学习完整的链路状态信息，便于路由更新管理。

默认情况下，在OSPF AS中的每个OSPF路由器必须在其LSDB（链路状态数据库）中保持有其他每个路由器的LSA（链路状态通告）。这样一来，在较大型的OSPF网络中，每个路由器中所保持的LSDB都比较大，进行SPF（最短路径优先）运算时需要消耗大量的系统资源。同时，所导致的直接结果就是路由表可能非常大，包含了到达AS中其他所有网络的路由。

2.理解Area

为了减小LSDB的大小，降低SPF运算的系统资源开销，减少路由表项数，OSPF允许AS被划分成多个连续网络群组，这就是“区域”（Area）。一个Area就是指OSPF路由域，或者AS的一部分，也就是说在一个AS中可以划分成多个区域。通过使用区域，可将大型网络划分成适合运行SPF（最短路径优先）路由算法的小块。图9-14所示的就是一个包含4个区域的OSPF网络，从中可以看出，各区域也是通过路由器进行相互连接的，此时这些路由器就是区域边界路由器，它们至少位于两个区域中。当然，在每个区域内部还会有路由器，这些就是前面所说的区域内部路由器。

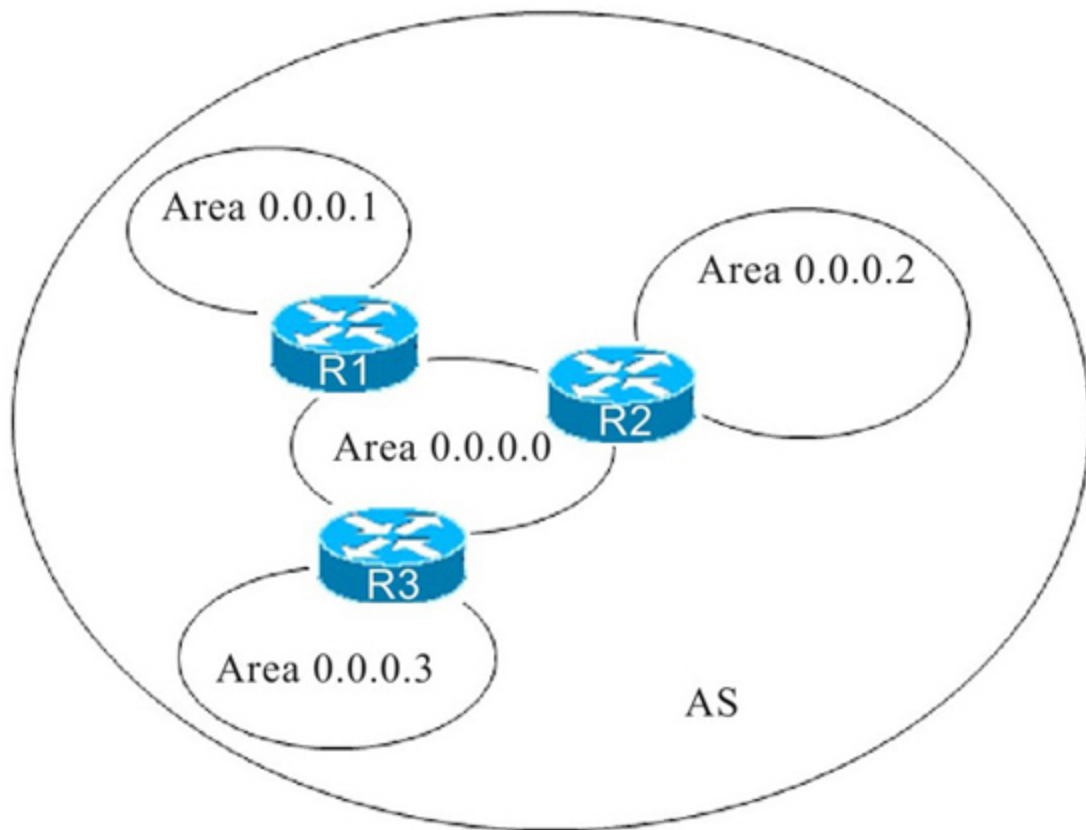


图 9-14 包含多个区域的OSPF网络示例

在OSPF路由协议中，每一个区域中的路由器都独立于计算网络拓扑结构，区域间的网络结构情况是互不可见的。每一个区域中的路由器也不会了解外部区域的网络结构，可谓是“各自为政，互不干涉”。这就意味着每一个区域都有着该区域独立的网络拓扑数据库及网络拓扑图。这样做的好处就是有利于减少网络中LSA报文在整个OSPF网络范围内的通告。

不同区域是通过一个四段、最多12位的十进制点分形式（与IP地址的表示形式一样）的区域ID来进行标识的，如1.1.1.1。但是，一个

ID仅是一个管理标识符，与IP地址或者IP网络ID没有任何关系。但是如果在一个AS中的一个区域中的各网段IP地址都是在一个网络或子网中，则出于管理上的便利，可以将区域ID设置为对应的网络或子网ID。例如，如果一个区域中所在网段都是在10.1.0.0这个网络之中，则区域ID可以设置为10.1.0.0。

为了使每个路由器的LSDB最小，一个区域中的网络LSA报文只向区域内部的路由器泛洪，而不会向区域外的路由器泛洪。每个区域以自己的LSDB构成自己的链路状态域。如果一个路由器连接了多个区域，则它有多个LSDB和SPF树。它的路由表就是该路由器所有SPF树中的路由表项的组合。为了减少OSPF路由器路由表中的路由条目数量，在区域内部的网络可以使用汇总路由通告（Summary-LSA）向区域外通告，也就是一个多个子网的LSA以一个大的网络LSA形式进行通告。

9.2.3 OSPF网络路由器类型

前面说了，在OSPF网络中，不仅有AS之分，在同一个AS内部又可以划分多个不同的区域。当一个AS划分成几个OSPF区域时，根据该路由器在相应的区域之内或者区域之间，甚至不同OSPF AS之间作用的不同，可以将OSPF路由器做如下分类：

(1) 内部路由器（Internal Router，IR）

当一个OSPF路由器上所有直连的链路（也就是路由器上所有接口）都处于同一个区域（不直接与其他区域相连）时，我们称这种路由器为内部路由器。内部路由器上仅仅运行其所属区域的OSPF运算法则，仅生成区域内部的路由表项。

(2) 区域边界路由器（Area Border Router，ABR）

当一个路由器有多个接口，其中至少有一个接口与其他区域相连时，我们称之为“区域边界路由器”。区域边界路由器的各对应接口运行与其相连区域定义的OSPF运算法则，具有相连的每一个区域的网络结构数据，并且了解如何将该区域的链路状态信息通告至骨干区域，再由骨干区域转发至其余区域。有关OSPF的区域分类将在本节后面各小节中介绍。

(3) AS边界路由器 (Autonomous System Boundary Router, ASBR)

AS边界路由器是与AS外部的路由器互相交换路由信息的OSPF路由器。该路由器在AS内部通告其所得到的AS外部路由信息，这样AS内部的所有路由器都知道AS边界路由器的路由信息。AS边界路由器的定义与前面几种路由器的定义是相互独立的，一个AS边界路由器可以是一个区域内部路由器，或是一个区域边界路由器。

(4) 骨干路由器 (Backbone Router)

骨干路由器是指至少有一个接口定义为属于骨干区域的路由器。任何一个与骨干区域互连的ABR或者ASBR也将成为骨干路由器。有关OSPF骨干区域也将在后面介绍。

以上四类OSPF网络路由器所处的位置如图9-15所示。从中可以看出它们之间的关系。

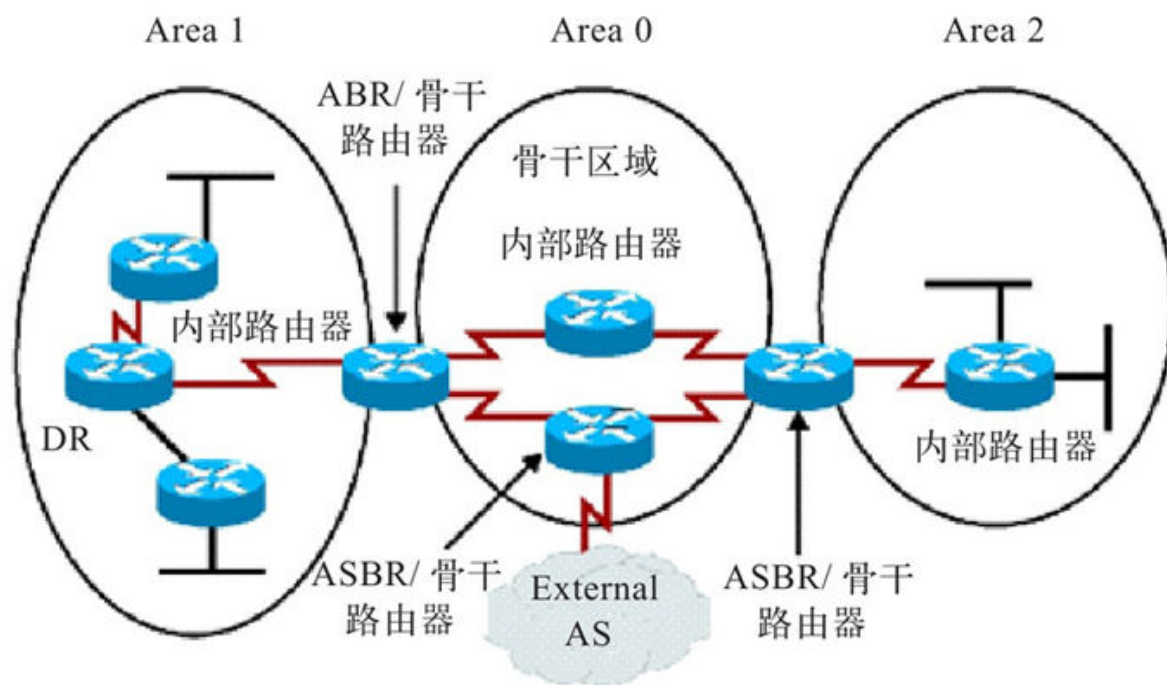


图 9-15 OSPF网络的主要路由器类型示例

9.2.4 DR和BDR

在一个广播性、多路访问的网络（例如Ethernet、TokenRing及FDDI环境）中，如果每个路由器都独立地与其他路由器进行LSU（Link State Update Packet，链路状态更新包）交换，以同步各自的LSDB，将导致一个巨大的流量增长。为了防止出现这种现象，同时使路由器保存的链路状态信息最少，OSPF在这类网络上选举出一个DR（Designated Router，指定路由器）和BDR（Backup Designated Router，备份指定路由器）。区域内那些既不是DR，也不是BDR的路由器称为DR Other。

1.DR和BDR的主要作用

DR就是集中负责一个区域内各路由器间的LSU交换和邻接关系建立，相当于这个区域的负责人一样。由于OSPF路由器之间是通过建立邻接关系及以后的泛洪来同步链路状态数据库的，所以DR必须与所有同一区域的OSPF路由器建立邻接关系（其他路由器间不必建立邻接关系），负责集中管理、维护和组播下发区域内各路由器发来的链路状态信息。DR或者BDR通常是处于一个区域的中心地位，使其他路由器与它建立邻接关系的难易程度相当。BDR用于在DR失效后接替DR的工作（也可不选举BDR），在DR正常工作时，它不承担DR的职责。在同一个OSPF区域中，每个路由器都和DR、BDR相连。

就像一个团体，其中必须要指定一个负责人。这个负责人负责其他成员的信息反馈、转发和广播（保存有所有成员之间的联系信息），其他每个成员也都与这个负责人建立单线联系，而其他成员之间彼此不建立联系。这样就不需要每个成员都保存整个团体会员的联系信息，也不必每个成员都与其他成员建立联系，管理更加简单。这个团体的负责人就相当于这里所说的DR了。

当区域中的路由器有路由更新时，DR Other路由器不会向其他DR Other路由器发送自己的LSU，只会向224.0.0.6这个组播地址发送，然后由224.0.0.6这个地址把DR Other发来的LSU组播给DR/BDR。在DR/BDR收到从224.0.0.6发过来的LSU后又会把这些LSU发给224.0.0.5这个组播地址，224.0.0.5再把LSU泛洪到区域内的所有DR Other路由器上。这样，区域网络上的所有DR Other路由器都会知道这个起源的DR Other路由器发送的路由更新，而不用向每个DR Other路由器都发送一次路由更新。

在图9-16所示的示例中，假设一开始没有选举DR和BDR，则在其中的任何一台路由器的路由发生了改变时，区域内的其他路由器都得重新进行SPF计算，生成新的LSDB表项。但如果选举了R3作为DR，R4和R5均为DR Other，则当R4所连接的网络发生变化时，R4只会把自己的LSU发送到224.0.0.6这个组播地址，然后224.0.0.6会把这个LSU传送给R3。R3在收到R4的LSU后会对比自己的LSDB，发现LSDB里面只

有关于R4的条目需要更新，于是便把一个关于R4的LSU发送到224.0.0.5组播地址，这时224.0.0.5就会在整个区域中组播R4的这一个LSU包。R5收到224.0.0.5发过来LSU之后直接更新自己LSDB，不需要重新进行SPF计算。

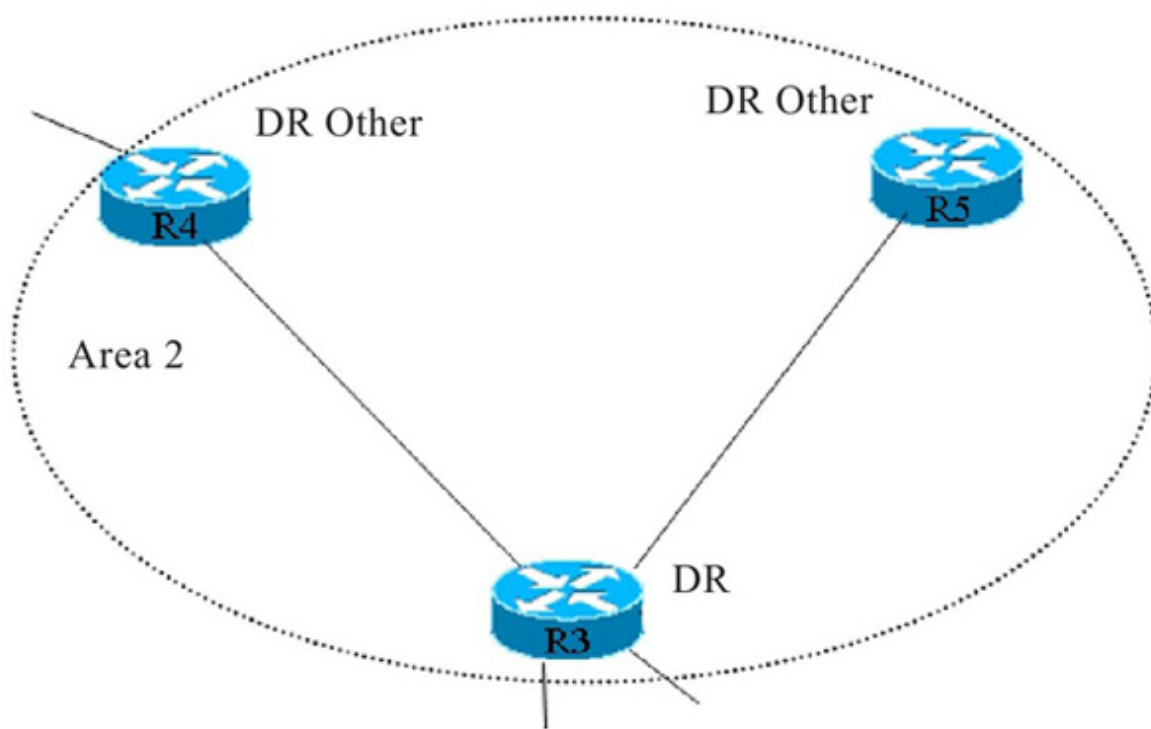


图 9-16 DR应用示例

通过这样一种LSU发送方式，就变原来的“多点对多点”信息发送方式为“单点到多点”信息发送方式。这样有两个好处：一是可更有效地利用网络带宽资源；二是可提高LSU信息发送效率。

DR Other仅与DR和BDR之间建立邻接关系，DR Other之间不交换任何路由信息。这样就减少了广播网和NBMA网络上各路由器之间邻接关系的数量，同时减少了网络流量，节约了带宽资源。如图9-17所示，用实线代表以太网物理连接，虚线代表建立的邻接关系。可以看到，采用DR/BDR机制后，5台路由器之间只需要建立7个邻接关系就可以了，如果不选举DR和BDR的话，大家画一下，看有多少条邻接关系。

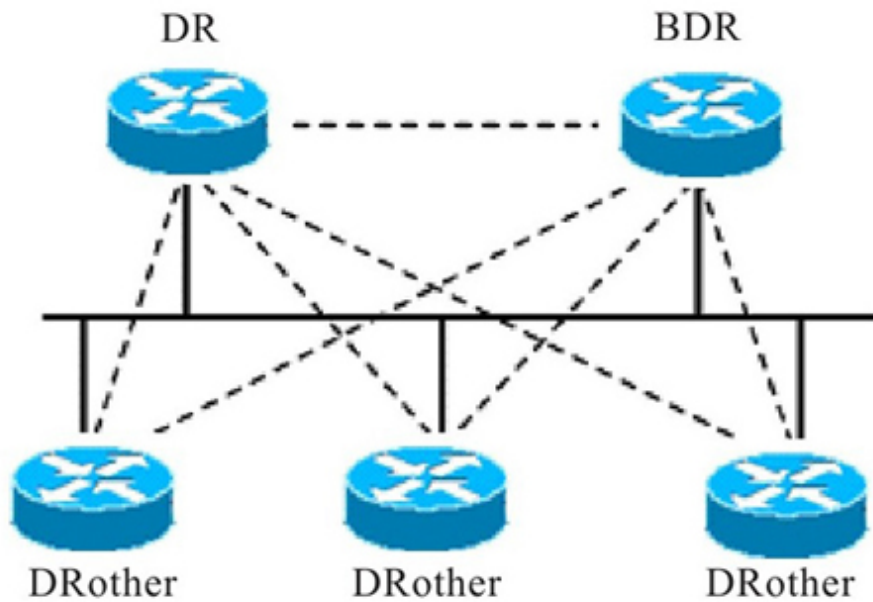


图 9-17 DR和BDR示意图

2.DR和BDR的选举

DR和BDR也不是随便担当的，需要选举产生，就像我们在学校选正、副班长一样，也是要投票选举的。DR是通过接口优先级

（Interface Priority）进行选举，最高优先级的路由器被选为DR，次高者被选为BDR；如果接口优先级相同，就按router-id进行选举，由最大到次大选举DR、BDR。

这里所说的接口优先级是由OSPF协议定义的（如Cisco路由器配置接口优先级命令是ip ospf priority），可以配置的优先级值在0~255之间（默认值为1，值越大，优先级越高）。但DR、BDR的选举不支持抢占（Preemption），也就是一旦选举完成，即使新加一个优先级更高的设备也不会进行重新的选择，相反只有在DR或者BDR出问题的时候才会发生重选。就像一个班上已选举了正、副班长，当新插入了一个学生后，也不会重新选举班长，即使新来的学生各方面能力比原班长、副班长强，除非原来的正、副班长出了问题。

9.2.5 OSPF LSA类型

OSPF是一种典型的链路状态路由协议，默认情况下采用OSPF的每个路由器彼此交换并保存整个网络的链路信息，从而掌握全网的拓扑结构，并独立计算路由。划分区域后，OSPF路由器收集其所在网络区域上各路由器的连接状态信息，即链路状态信息，生成LSDB。然后OSPF路由器利用SPF路由算法独立地计算出到达任意目的地的路由。

随着OSPF路由器类型概念的引入，OSPF路由协议又对其链路状态通告（LSA）数据包作出了分类。OSPF将链路状态通告数据包共分成以下7类。

（1）Type 1: 路由器LSA（Router LSA）

路由器LSA（第一类通告）可由区域内所有路由器产生，并且只能在本在区域内泛洪通告。路由器LSA描述了路由器物理接口所连接的链路或接口，指明了链路的状态、开销等参数。第一类LSA只在产生的区域内泛洪。

（2）Type 2: 网络LSA（Network LSA）

网络LSA（第二类通告）是由DR（指定路由器）或者BDR（备份指定路由器）产生的，报文包括DR和BDR连接的路由器的链路信息，描述了一个多路访问网络所有相连的路由器。第二类的LSA也只在产生的区域内泛洪。

（3）Type 3: 网络汇总LSA（Network summary LSA）

网络汇总LSA（第三类通告）是由ABR产生的，报文包通知本区域内的路由器通往区域外的路由信息。在一个区域外部但是仍然在一个OSPF自治系统内部的默认路由也可以通过这种LSA来通告。

如果一台ABR路由器经过骨干区域从其他的ABR路由器收到多条网络汇总LSA，那么这台始发的ABR路由器将会选择这些LSA通告中代价最低的LSA，并且将这个LSA的最低代价通告给与它相连的非骨干区域。

（4）Type 4: ASBR汇总LSA（ASBR summary LSA）

ASBR汇总（第四类通告）LSA也是由ABR发出的，但它是一条主机路由，即指向ASBR路由器地址的路由。

（5）Type 5: 自治系统外部LSA（Autonomous system external LSA）

自治系统外部LSA（第五类通告）是由ASBR产生的，告诉相同自治系统的路由器通往外部自治系统的路径。自治系统外部LSA是唯一不和具体的区域相关联的LSA通告，将在整个自治系统中进行泛洪。

（6）Type 6: 组成员LSA（Group membership LSA）

第六类通告应用很少，且目前不支持组播OSPF（MOSPF）协议，这类通告本处不做具体介绍。

（7）Type 7: NSSA外部LSA（NSSA External LSA）

NSSA外部LSA（第七类通告）是由ASBR产生的，内容和LSA 5几乎是相同的，但NSSA外部LSA通告仅在始发这个NSSA外部LSA通告的非纯Stub区域内部进行泛洪。

在NSSA区域中，当有一个路由器是ASBR时，不得不产生LSA 5报文，但是NSSA中不能有LSA 5报文，所有ASBR产生LSA 7报文，发给本区域的路由器。只有一个例外，每台ABR路由器利用一个类型3来通告默认路由。

Opaque LSA是一个被提议的LSA类别，在标准的LSA头部后面加上特殊应用的信息组成，可以直接由OSPF协议使用，或者由其他应用分发信息到整个OSPF域间接使用。Opaque LSA分为Type9、Type10、Type11三种类型，但它们各自可泛洪的区域不同：其中，Type9的

Opaque LSA仅在本地链路范围进行泛洪，Type10的Opaque LSA仅在本地区域范围进行泛洪，Type11的LSA可以在一个自治系统范围进行泛洪。

9.2.6 Backbone（骨干）区域

在一个大型的OSPF网络中，可以包括多种区域，其中就有三种常见的特殊区域，即骨干区域（Backbone Area）、末梢区域（Stub Area）和非纯Stub区域（No Stotal Stub area, NSSA），当然还可以包括其他普通区域。OSPF网络中的区域是以区域ID进行标识的，区域ID为0的区域规定为骨干区域。这几类区域如图9-18所示。本节仅介绍其中的骨干区域，其它类型区域将在本章后面介绍。

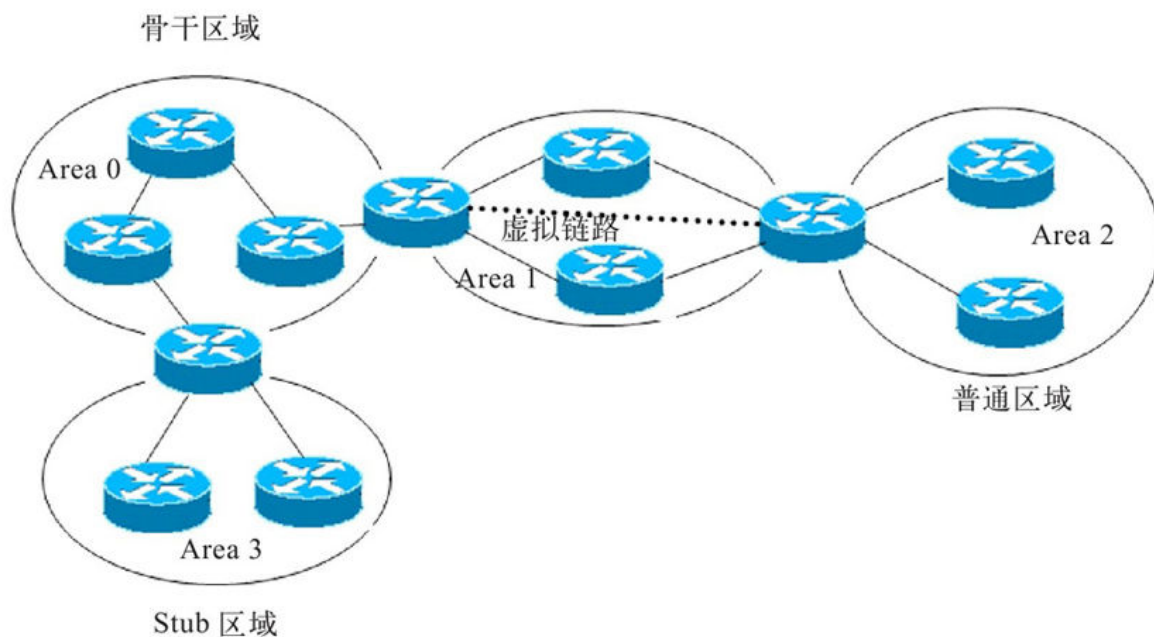


图 9-18 OSPF主要区域类型示例

一个OSPF互联网络，无论有没有划分区域，总是至少有一个骨干区域。骨干区域有一个ID为0.0.0.0，称为区域0。另外，骨干区域必须是连续的（也就是中间不会越过其他区域），其余区域必须与骨干区域直接相连（但事实上，有时并不一定会这样，所以也就有了下面将要介绍的“虚拟链路”技术）。骨干区域一般为区域0（Area 0），其主要工作是在其余区域间传递路由信息。

骨干区域作为区域间传输通信和分布路由信息的中心，区域间的通信先要被路由到骨干区域，然后再路由到目的区域，最后被路由到目的区域中的主机。在骨干区域中的路由器通告他们区域内的汇总路由到骨干区域中的其他路由器。这些汇总通告在区域内路由器泛洪，所以区域中的每台路由器都有一个反映其所在区域内路由可用的路由表，这个路由与AS中其他区域的ABR汇总通告相对应。

如在图9-14所示的OSPF网络中，R1使用一个汇总通告向所有骨干路由器（R2和R3）通告Area 0.0.0.1中的所有路由。R1从R2和R3接收汇总通告。R1配置了Area 0.0.0.0中的汇总通告信息，通过泛洪，R1把这个汇总路由信息传播到Area 0.0.0.1内所有路由器上。在Area 0.0.0.1内的每个路由器，依据来自Areas 0.0.0.0、0.0.0.2和0.0.0.3区域的汇总路由信息完成路由表的计算。

在实际网络中，可能会存在骨干区域不连续，或者某一个区域与骨干区域物理不相连的情况，此时系统管理员可以通过设置虚拟链路

（Virtual Link）的方法来解决该问题（参见图9-18中Area 1中的“虚拟链路”）。虚拟链路存在于两个路由器之间，这两个路由器都有一个端口与同一个非骨干区域（这个区域是处于骨干区域和某个不直接与骨干区域相连的区域之间）相连，虚拟链路使该区域与骨干区域间建立一个逻辑连接点。

虚拟链路被认为属于骨干区域（相当于骨干区域的延伸），在OSPF路由协议看来，虚拟链路两端的两个路由器被一个点对点的链路连在一起，这样原来本来没有与骨干区域连接的区域就变成直接连接，成为普通区域了。而且，在OSPF路由协议中，通过虚拟链路的路由信息是作为域内路由来看待的。但是，该虚拟链路必须建立在两个区域边界路由器之间，并且其中一个区域边界路由器必须属于骨干区域。

9.2.7 Stub（末梢）区域

通过前面对OSPF区域概念的了解可以知道，在划分了区域之后，OSPF网络中的非骨干区域中的路由器要路由到外部区域，一定要通过ABR（区域边界路由器）来转发。或者说对于区域内的路由器来说，ABR是一个通往外部区域的必经之路。既然如此，对于区域内的路由器来说，就没有必要知道通往外部区域的详细路由了，只要由ABR向该区域发布一条默认路由来告知报文的发送路径即可。这样在区域内的路由器中就只需要为数不多的区域内路由和一条指向ABR的默认路由即可，使区域内的路由表简化。而且无论区域外的路由如何变化，都不会影响到区域内路由器这个简单的路由表。这就是OSPF路由协议中“Stub Area”（末梢区域）的设计理念。

Stub区域是一种比较特殊的区域，该区域的ABR不能接收其他OSPF AS路由，因为该区域中不存在ASBR。在Stub区域的内部路由器仅需要配置一条到达该区域ABR的默认路由（0.0.0.0 0.0.0.0）来实现同一AS中不同区域间的路由，这样可使得这些区域中内部路由器的路由表规模以及路由信息传递的数量都会大大减少。但要注意的是，并不是每个区域都可以配置为Stub区域，也就是说要配置为Stub区域还需符合一定的条件。具体来说配置为Stub区域具有以下特点：

□Stub区域位于AS的边界，是那些只有一个ABR的非骨干区域。为保证到AS内其他区域的路由依旧可达，该区域的ABR将生成一条默认路由，并发布给Stub区域中的其他非ABR路由器。

□骨干区域不能配置成Stub区域。

□Stub区域内不能存在ASBR，即自治系统外部的路由不能在该区域内传播。

□虚连接不能穿过Stub区域。

在Stub区域中的各路由器上所创建的默认路由是用于Stub区域中各路由器为任一不可到达AS内部的外部IP地址提供的唯一路由。在Stub区域中的所有路由器必须被配置为该默认路由，以便它们不在Stub区域内导入或泛洪AS外部路由。所以，在一个Stub区域中的所有路由器接口上的所有区域配置必须配置Stub区域。例如，图9-19中的Area 0.0.0.3被配置为一个Stub区域，因为所有外部通信必须通过它的单个ABR——R3到达。R3通告一个默认路由分布在内部区域Area 0.0.0.3，但不会在区域内泛洪AS外部区域路由。

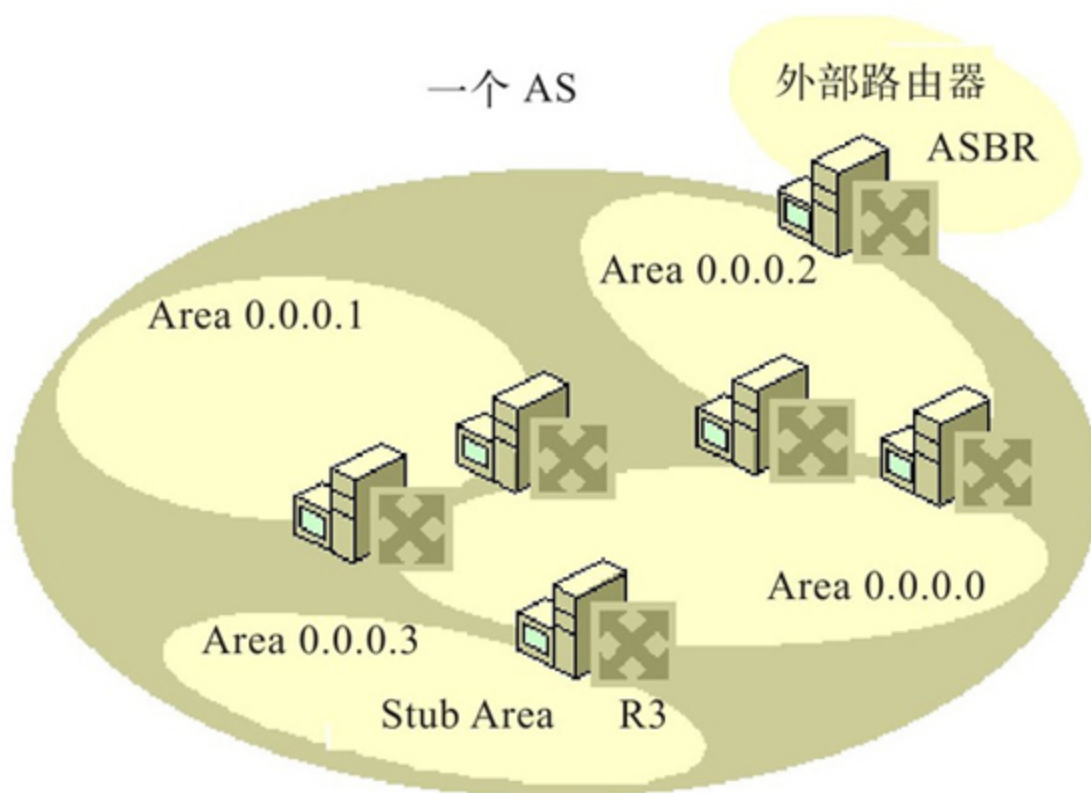


图 9-19 OSPF Stub区域示例

由于Stub区域通常位于OSPF网络末端，这些区域内的路由器通常是由一些处理能力有限的低端路由器组成，所以处于Stub区域内的这些低端设备既不需要保存庞大的路由表，也不需要经常性地路由计算。这样做有利于减小Stub区域中内部路由器上的链路状态数据库的大小及存储器的使用，提高路由器计算路由表的速度。

当一个OSPF的区域只存在一个区域出口点（只与一个其他区域连接）时，我们可以将该区域配置成一个Stub区域。这时，该区域的边界路由器会对域内通告默认路由信息。需要注意的是，一个Stub区域

中的所有路由器都必须知道自身属于该区域（也就是需要在其中的路由器中启用这项功能），否则Stub区域的设置不会起作用。

9.2.8 Totally Stub区域和NSSA区域

上节介绍的Stub区域是一类特殊的OSPF区域，这类区域不接收或扩散Type 5类型LSA（AS外部LSA），对于产生大量Type 5 LSA的网络，这种处理方式能够有效减小Stub区域内路由器的LSDB大小，并缓解SPF运算对路由器资源的占用。通常情况下，Stub区域位于自治系统边缘区域。为保证Stub区域去往自治系统外的报文能被正确转发，Stub区域的ABR（区域边界路由器）将通过Summary-LSA（汇总链路状态通告）向本区域内发布一条默认路由，并且只在本区域泛洪。为了进一步减少Stub区域中路由器的路由表规模以及路由信息传递的数量，可以将该区域配置为Totally Stub（完全末梢）区域，该区域的ABR不会将区域间的路由信息和外部路由信息传递到本区域。

1. 完全Stub区域

这里所说的完全Stub区域（Totally Stub，或者Stub no-summary）是在Stub区域的基础上（即阻止了Type 5 LSA包的基础上）再阻止其他ABR通告的网络汇总LSA（即Type 3类型LSA），不接收区域间路由通告。其ABR仅通过网络汇总LSA通告一个默认路由，使用这个默认路由可到达OSPF自治系统外部其他区域。也就是说，完全Stub区域同时不允许Type 3、4或5三类LSA注入，但默认汇总路由除外。

2.NSSA区域

Stub区域虽然为合理地规划网络描绘了美好的前景，但在实际的组网中利用率并不高（Stub区域一般只存在于网络边缘），未免遗憾。但此时的OSPF协议已经基本成型，不可能再做大的修改。为了弥补缺陷，协议设计者提出了一种新的概念NSSA（not-so-Stubby area，非纯末梢区域），并且作为OSPF协议的一种扩展属性单独在RFC 1587中描述。NSSA对原来的Stub区域要求有所放宽，使它可以在更多网络环境中得到应用。

NSSA区域规定，AS外的ASE（AS外部）路由不可以进入到NSSA区域中，但是NSSA区域内的路由器引入的ASE路由（NSSA区域中可以连接ASBR）可以在NSSA中泛洪并发送到区域之外。这样，在NSSA区域中取消了原来Stub区域中关于ASE的双向传播的限制（区域外的进不来，区域里的也出不去），改为单向限制（区域外的进不来，区域里的能出去）。

图9-20所示是一个包含NSSA区域的OSPF网络示例。NSSA所对应的区域ID为1，与骨干区域0相连；同时它又与IGRP和RIP路由协议这两个不同的AS相连。此时，外部区域和外部AS的路由信息不能通告到NSSA区域中，但NSSA中的路由信息可以向外发布。

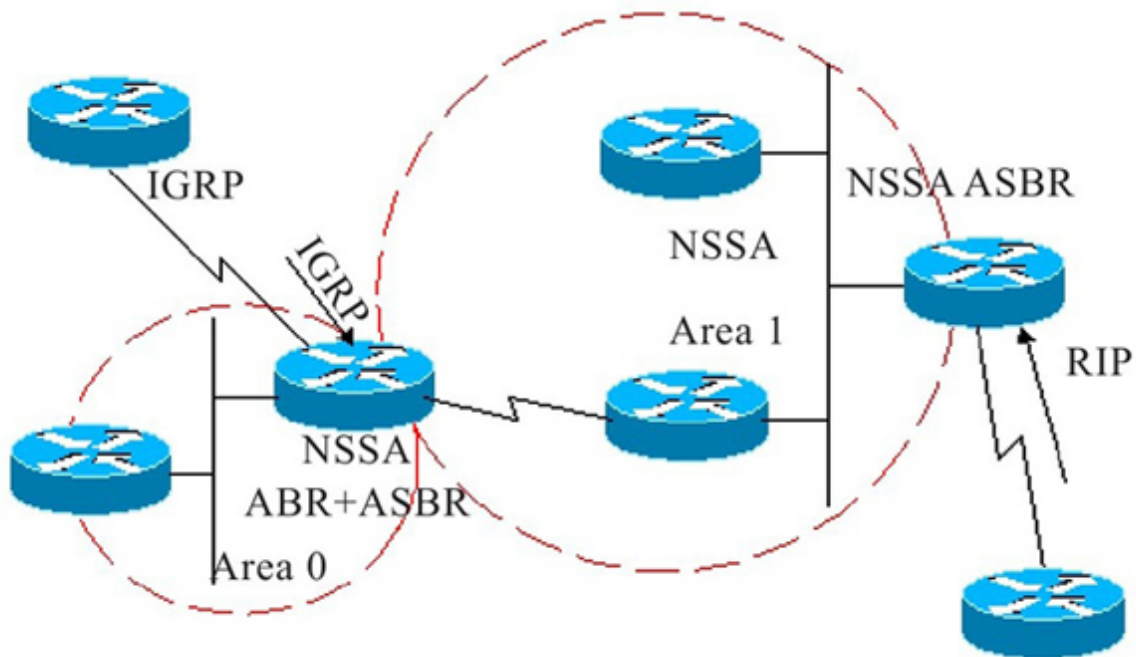


图 9-20 OSPF NSSA区域示例

为了解决ASE单向传递的问题，NSSA中重新定义了一种LSA——Type 7类型的LSA（NSSA外部LSA），供区域内的路由器引入外部路由时使用。该类型的LSA除了类型标识与Type 5不相同之外，其他内容基本一样。这样区域内的路由器就可以通过LSA的类型来判断是否该路由来自本区域内。但由于Type 7类的LSA是新定义的，对于不支持NSSA属性的路由器无法识别，所以协议规定：在NSSA的ABR上将NSSA内部产生的Type 7类型的LSA转化为Type 5类型的LSA再发布出去，并同时更改LSA的发布者为ABR自己。这样NSSA区域外的路由器就可以完全不用支持该属性。在NSSA区域内的所有路由器（包括NSSA的ABR）必须支持Type 7类型的LSA属性，而自治系统中的其他路由器则不需要。

总的来说，NSSA区域不允许Type 5 LSA，但在NSSA ABR上转换为Type 5的Type 7 LSA还是可以通过的。

9.2.9 OSPF路由计算基本过程

在OSPF网络，路由的计算不是简单地把源地址与目的地址进行关联，需要考虑到许多因素，以确定一条最佳路径。整个OSPF路由计算过程可分为：邻接关系建立→DR/BDR选举→发送LSA→创建路由表→维护路由表这五大基本步骤。具体描述如下。

(1) 建立邻接关系

所谓“邻接关系”（Adjacency）是指OSPF路由器以交换路由信息为目的，在所选择的相邻路由器之间建立的一种关系。在OSPF中，邻居（Neighbor）和邻接（Adjacency）是两个不同的概念。OSPF路由器启动后，便会通过OSPF接口向外发送Hello报文。收到Hello报文的OSPF路由器会检查报文中所定义参数，如果双方一致就会形成邻居关系。但形成邻居关系的双方不一定都能形成邻接关系，这要根据网络类型而定。只有当双方成功交换DD（Database Description，数据库描述）报文，交换LSA并达到LSDB的同步之后，才形成真正意义上的邻接关系。

具体步骤是：路由器首先发送拥有自身ID信息（Loopback端口或最大的IP地址）的Hello报文。与之相邻的路由器如果收到这个Hello报文，就将这个报文内的ID信息加入到自己的Hello报文内。然后在后面

发送的Hello报文中就包括了原来所接收到的邻居路由器的ID信息。如果路由器的某端口收到从其他路由器发送的含有自身ID信息的Hello报文，则它根据该端口所在网络类型确定是否可以与对端路由器建立邻接关系。

在点对点网络中，路由器将直接和对端路由器建立起邻接关系，并且该路由器将直接进入下面的第3步，发送LSA以发现其他路由器；若为多路访问网络，则该路由器将进入下面第2步。

(2) 选举DR/BDR

在广播或者多路访问OSPF网络中，各相邻路由器都建立了相邻关系后，就要选举一个担当区域内的LSU通告代理角色的DR（指定路由器）和BDR（备份指定路由器），因为在OSPF网络中，为了减少LSU通告的流量，各路由器之间不直接发送链路状态信息，而是通过选举DR/BDR进行统一分发。其他路由器要发送LSU，则先把LSU发给DR/BDR，再由DR或者BDR（只有在DR失效时才使用它）在组播给所有非DR或者BDR的路由器。

DR和BDR是由同一网段中所有的路由器根据路由器优先级、Router ID通过Hello报文选举出来的，只有优先级大于0的路由器才具有选举资格。具体的选举过程如下：

1) 在与一个或多个邻居之间的双向通信建立起来之后，本地路由器对每个邻居发送来的Hello包中的优先级、DR和BDR域进行检查。此时所有路由器都宣称自己为DR（将它们自己的接口地址置于Hello包的DR域中）；而且所有路由器都宣称自己为BDR（将它们自己的接口地址置于Hello包的BDR域中）。

2) 如果一或多个备选路由器将它（们）自身的接口地址置于DR域中，拥有最高优先级的邻居将被宣告为DR。如果路由器优先级一样，拥有最高Router ID的邻居将被选举出来。

3) 然后在将自身的接口地址置于BDR域中的路由器中选择拥有最高优先级的路由器作为BDR。如果这些宣称自己为BDR路由器的优先级相等，则拥有最高Router ID的邻居将被选举作为BDR。

4) 如果没有任何路由器被宣告为BDR，则拥有最高优先级的非DR邻居路由器将被选举为BDR；如果多个这样的路由器优先级相同，则拥有最高Router ID的邻居将被选举为BDR。

(3) 发送LSA

作为一种典型的链路状态的路由协议，OSPF还得遵循链路状态路由协议的统一算法。当路由器初始化或当网络结构发生变化（例如增减路由器，链路状态发生变化等）时，路由器会产生链路状态广播（LSA）数据包，该数据包里包含路由器上所有相连链路，也即为所

有端口的状态信息。所有路由器会通过泛洪方式来交换链路状态数据。

在这步，路由器与路由器之间首先利用Hello报文的ID信息确认主从关系，然后主从路由器相互交换部分链路状态信息。每个路由器对信息进行分析比较，如果收到的信息有新的内容，路由器将要求对方发送完整的链路状态信息。这个状态完成后，路由器之间建立完全邻接关系，同时各邻接路由器拥有自己独立的、完整的链路状态数据库。

在多路访问网络内，DR与BDR互换信息，并同时与本子网内其他路由器交换链路状态信息。在点对点或点对多点网络中，相邻路由器之间会直接交换链路状态信息。

(4) 创建路由表

当网络重新稳定下来，也可以说OSPF路由协议收敛下来时，所有的路由器会根据其各自的链路状态信息数据库采用SPF（最短路径优先）算法计算并创建路由表。OSPF路由器依据链路状态数据库的内容，独立地用SPF算法计算出到每一个目的网络的路径，并将路径存入路由表中。该路由表中包含路由器到每一个可到达目的地的开销以及到达该目的地所要转发的下一个路由器（next-hop）。

OSPF利用开销来计算路由路径性能的，开销最小者即为最短路径。在配置OSPF路由器时可根据实际情况，如链路带宽、时延等设置链路的开销大小；开销越小，则该链路被选为路由的可能性越大。这里的开销是根据链路类型来计算的，不同的链路类型对应的开销值不一样。

(5) 维护路由信息

当链路状态发生变化时，OSPF通过泛洪过程广播网络上的其他路由器。OSPF路由器接收到包含有新信息的链路状态更新报文，将更新自己的链路状态数据库，然后用SPF算法重新计算路由表。在重新计算过程中，路由器继续使用旧路由表，直到SPF完成新的路由表计算。新的链路状态信息将发送给其他路由器。值得注意的是，即使链路状态没有发生改变，OSPF路由信息也会自动更新，默认时间为30min。

9.2.10 OSPF报头格式

OSPF报文直接封装为IP协议报文，因为OSPF是专为TCP/IP网络而设计的路由协议。OSPF报文主要有5种：Hello报文、DD（Database Description，数据库描述）报文、LSR（Link State Request，链路状态请求）报文、LSU（Link State Update，链路状态更新）报文和LSAck（Link State Acknowledgment，链路状态应答）报文。它们使用相同的OSPF报头格式，如图9-21所示。

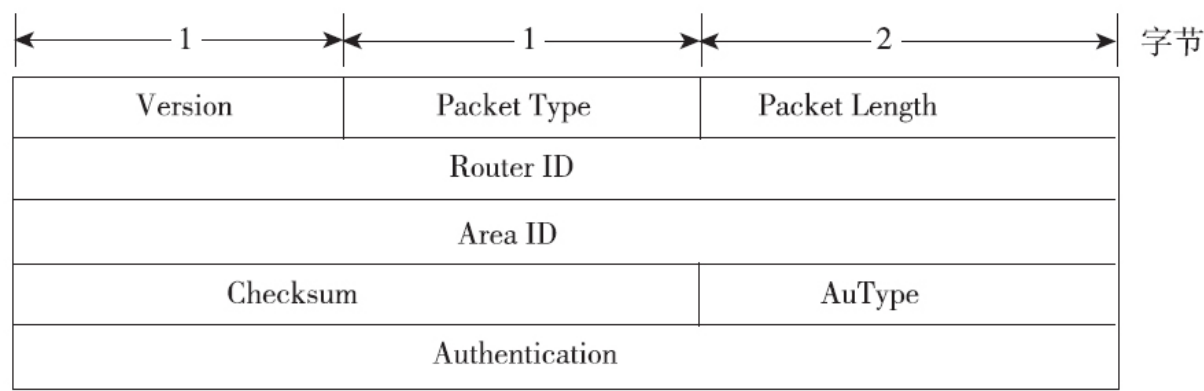


图 9-21 OSPF协议报头格式

(1) Version

版本字段，占1字节，指出所采用的OSPF协议版本号，目前最高版本为OSPF v4，即值为4（对应二进制就是0100）。

(2) Packet Type

报文类型字段，标识对应报文的类型。前面说了OSPF有5种报文，分别是：Hello报文、DD报文、LSR报文、LSU报文、LSAck报文。这五种类型字段解释如下。

1) Hello报文：OSPF Hello报文是用来发现OSPF邻居并维持OSPF邻居关系的报文。Hello报文被周期地发向邻居路由器接口发送，报文内容包括一些定时器设置、DR、BDR以及本路由器已知的邻居路由器。

整个Hello报文格式如图9-22所示，上部分为图9-21所示的OSPF报头部分，下部分为Hello报文内容部分。Hello报文内容部分各字段说明如表9-1所示。

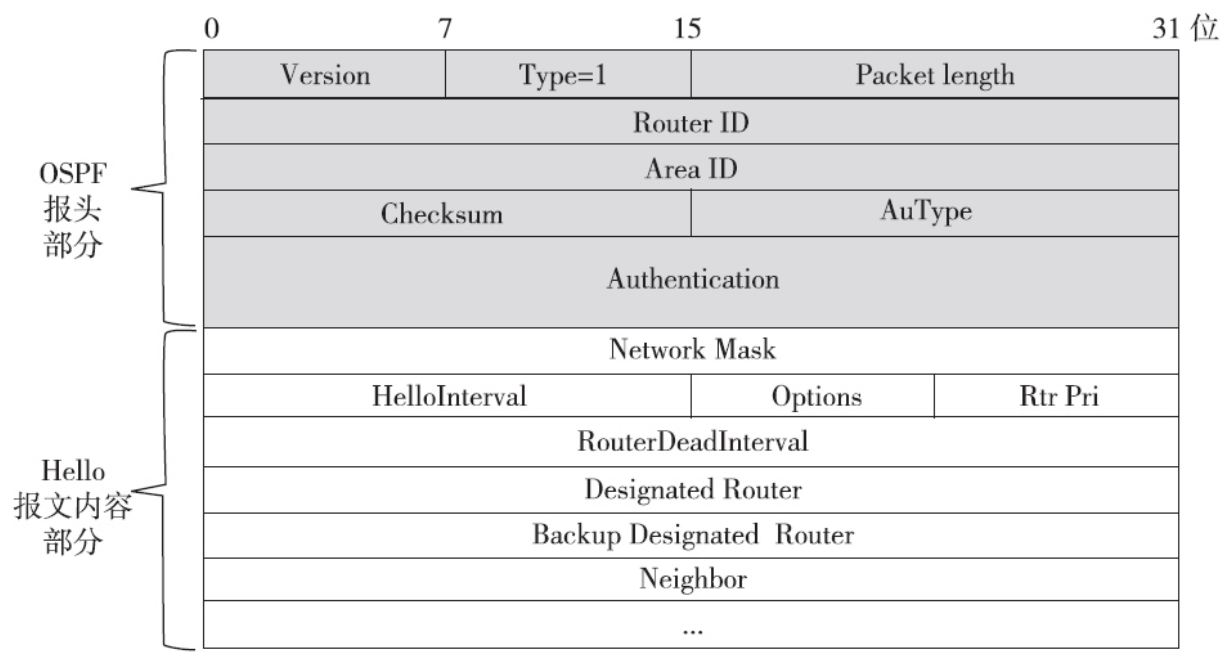


图 9-22 Hello报文格式

表 9-1 Hello 报文内容部分字段说明

字段名	长度 / 字节	功 能
Network Mask	4	发送 Hello 报文接口所在的子网掩码
HelloInterval	2	指定发送 Hello 报文的时间间隔，默认为 10s
Options	1	可选项，包括 E，允许泛洪 AS-external-LAS；MC，允许转发 IP 组播报文；N/P，允许处理 Type 7 LSA；DC，允许处理按需链路
Rtr Pri	1	指定 DR 优先级，默认为 1。如果设为 0，则表示本路由器不参与 DR/BDR 选举
RouterDeadInterval	4	指定路由器失效时间，默认为 40s。如果在此时间内没有收到邻居路由器发来的 Hello 报文，则认为该邻居路由器已失效
Designated Router	4	指定 DR 的接口 IP 地址

(续)

字段名	长度 / 字节	功 能
Backup Designated Router	4	指定 BDR 的接口 IP 地址
Neighbor	4	指定邻居路由器的 RID。下面的省略号 (...) 表示可以指定多个邻居路由器 RID

2) DD报文：DD报文是用来描述本路由器的链路状态数据库（LSDB），进行数据库同步的。DD报文内容部分包括DD报文序列号和LSDB中每一条LSA的头部等，如图9-23所示。各字段说明如表9-2所示。

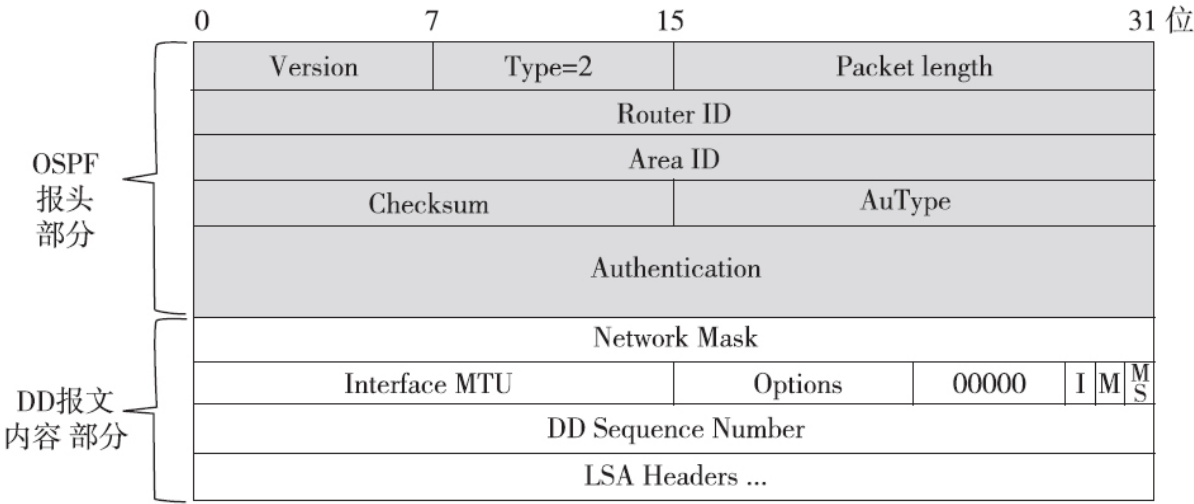


图 9-23 DD报文格式

表 9-2 DD 报文内容部分字段说明

字段名	长 度	功 能
Interface MTU	2 字节	指出发送 DD 报文的接口在不分段的情况下，可以发出的最大 IP 报文长度
Options	1 字节	可选项，包括 E: 允许泛洪 AS-external-LAS; MC, 允许转发 IP 组播报文; N/P, 允许处理 Type 7 LSA; DC, 允许处理按需链路
I	1 位	指定在连续发送多个 DD 报文，如果是第一个 DD 报文则置 1，其他的均置 0
M	1 位	指定在连续发送多个 DD 报文，如果是最后一个 DD 报文则置 0，否则均置 1
M/S	1 位	设置进行 DD 报文双方的主从关系，如果本端是 Master 角色，则置 1，否则置 0
DD Sequence Number	4 字节	指定所发送的 DD 报文序列号。主从双方利用序列号来确保 DD 报文传输的可靠性和完整性
LSA Header	4 字节	指定 DD 报文中所包括的 LSA 头部。后面的省略号 (…) 表示可以指定多个 LSA 头部

对端路由器根据所收到的DD报文中的OSPF报头就可以判断出是否已有这条LSA。在DD报文交换中，一台为Master（主）角色，另一台为Slave（从）角色。Master规定起始序列号，每发送一个DD报文，序列号加1，Slave则使用Master的序列号进行确定应答。

3) LSR报文：当两台路由器互相交换完DD报文后，知道对端路由器有哪些LSA是本LSDB所没有的，以及哪些LSA是已经失效的，则需要发送一个LSR报文，向对方请求所需的LSA。LSR报文内容包括所需的LSA摘要，具体格式如图9-24所示，LSR报文内容部分各字段说明如表9-3所示。

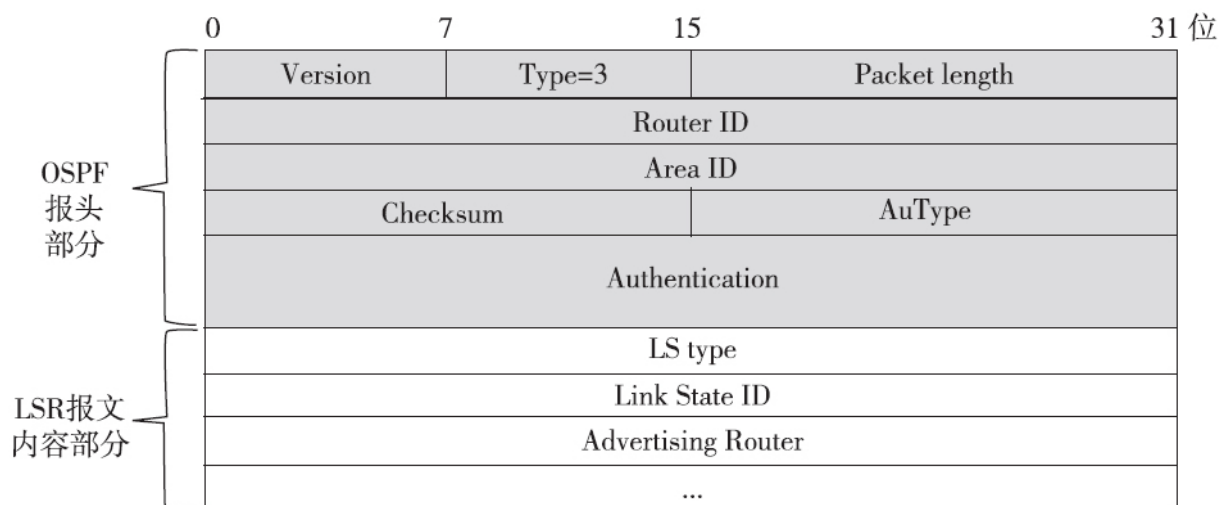


图 9-24 LSR报文格式

表 9-3 LSR 报文内容部分字段说明

字段名	长度 / 字节	功 能
LS type	4	指定所请求的 LSA 类型，主要共 7 类，具体参见 9.2.5 节
Link State ID	4	用于指定 ospf 所描述的部分区域，该字段的使用方法根据不同的 LSA 类型而不同：当为 LSA 1 时，该字段值是产生 LSA 1 的路由器的 Router ID；当为 LSA 2 时，该字段值是 DR 的接口地址；当为 LSA 3 时，该字段值是目的网络的网络地址；当为 LSA 4 时，该字段值是 ASBR 的 Router ID；当为 LSA 5 时，该字段值是目的网络的网络地址
Advertising Router	4	指定产生此请求 LSA 的路由器 ID

4) LSU报文: LSU报文是用来向对端路由器发送所需的LSA，内容是多条LSA完整内容的集合，LSU报文内容部分包括此次共发送的LSA数量和每条LSA的完整内容，如图9-25，报文内容部分的两个字段如表9-4所示。

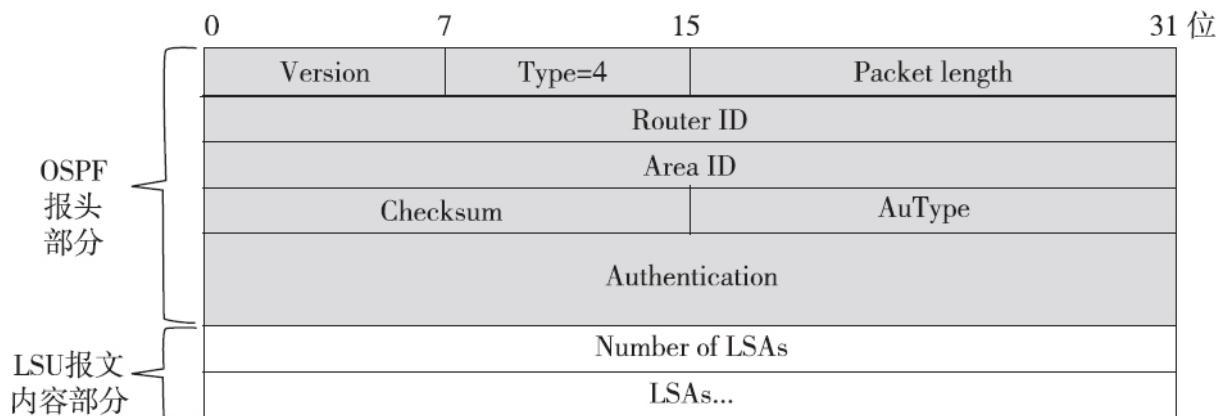


图 9-25 LSU报文格式

表 9-4 LSU 报文内容部分字段说明

字段名	长度 / 字节	功 能
Number of LSAs	4	指定此报文中共发送的 LSA 数量
LSAs	4	是一条条具体的 LSA 完整信息，后面的省略号表示可多条 LSA

LSU报文在支持组播和多路访问的链路上是以组播方式将LSA泛洪出去的，并且对没有收到对方确认应答（就是下面将要介绍的LSAck报文）的LSA进行重传，但重传时的LSA是直接送到没有收到确认应答的邻居路由器上，而不再是泛洪。

5) LSAck报文：这是路由器在收到对端发来的LSU报文后所发出的确认应答报文，内容是需要确认的LSA头部（LSA Headers），整个LSAck报文的格式如图9-26所示。LSAck报文根据不同链路以单播或组播形式发送。

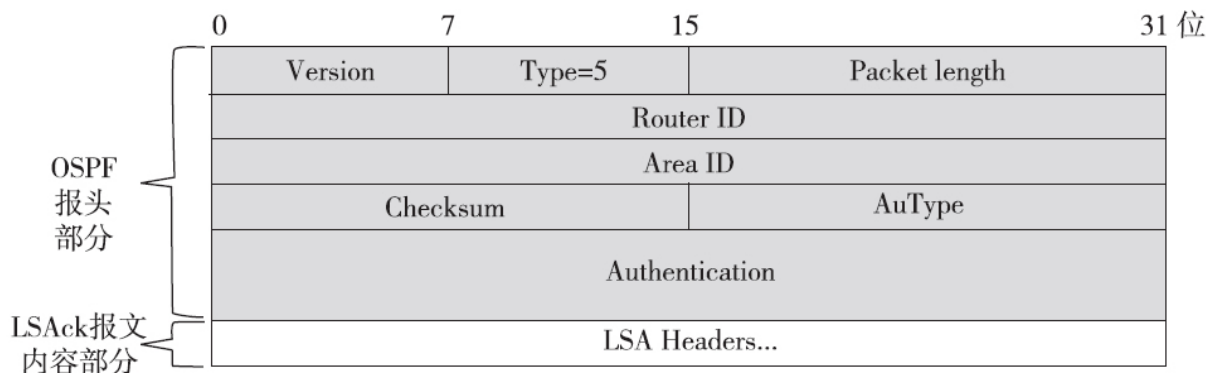


图 9-26 LSACK报文格式

下面继续介绍图9-21中所示的其他OSPF报头字段。

□**Packet Length**: 包长度字段，占2字节。它是指整个报文（包括OSPF报头部分和后面各报文内容部分）的字节长度。

□**Router ID**: 路由器ID字段，占4字节，指定发送报文的源路由器ID。

□**Area ID**: 区域ID字段，占4字节，指定发送报文的路由器所对应的OSPF区域号。

□**Checksum**: 校验和字段，占2字节，是对整个报文（包括OSPF报头和各报文具体内容，但不包括下面的Authentication字段）的校验和，用于对端路由器校验报文的完整性和正确性。

□**AuType**: 认证类型字段，占2字节，指定所采用的认证类型，0为不认证，1为进行简单认证，2采用MD5方式认证。

□**Authentication**: 认证字段，占8字节，具体值根据不同认证类型而定。认证类型为不认证时，此字段没有数据；认证类型为简单认证时，此字段为认证密码；认证类型为MD5认证时，此字段为MD5摘要消息。

9.3 IS-IS路由协议

IS-IS（Intermediate System-to-Intermediate System，中间系统到中间系统）是一个链路状态内部网关协议（Interior Gateway Protocol，IGP），是OSI/RM协议栈中CLNS（Connectionless Network Service，无连接网络服务）的一部分。我们知道，链路状态协议的特点就是信息的传播需要在每个参与路由协议的路由器上构建一个完整的网络互连映射。这个映射是用来计算到达目的节点或网络的最短路径的。

9.3.1 ISO网络基础

在OSI/RM体系结构的ISO网络中，定义了两类网络层服务：CLNS（Connectionless Network Service，无连接网络服务）和CONS（Connection-oriented Network Service，面向连接的网络服务）。提供CLNS服务的网络连接协议称为CLNP（Connectionless Network Protocol，无连接网络协议），类似于TCP/IP协议网络中的IP协议，都是无连接的网络层协议。提供CONS服务的网络连接协议称为CONP（Connection-oriented Network Protocol，面向连接网络协议），类似于Netware网络中的IPX协议。在整个ISO网络中，提供CLNS服务的有：CLNP、IS-IS和ES-IS，这三个分别实现不同的功能。在此对这三个协议进行简单介绍。

1.CLNP

CLNP是ISO网络层数据报协议，对于传输层来说，它提供了与TC/IP网络中的IP协议类似的基础功能，因此，CLNP又称ISO-IP。与IP协议一样，CLNP也是一个无连接的网络层协议，提供无连接的网络层服务。我们都知道，IP是TCP/IP协议栈中唯一的网络层协议，高层的协议和数据全都封装在IP数据包中，然后再传输到数据链路层，重新封装在帧中进行传输。而在ISO网络环境中，前面说的CLNP、IS-IS、ES-IS都是独立的网络层协议，都直接被封装到数据链路层的帧中进行传输。CLNP使用NSAP地址（IP协议使用的是IP地址）和NET（网络实体标识）来识别网络设备。

2.IS-IS

IS-IS最早仅在使用CLNP协议的ISO网络中实现各路由器间路由信息交换的协议，也就是ISO网络的动态路由协议。但是现在我们通常说IS-IS不仅适用于ISO网络，同样适用于TCP/IP网络，因为现在的IS-IS是经过标准化并且得到扩展后的版本。

IS-IS协议最早是在1980年后期由DEC公司开发的，后来由ISO以标准的形式发布在ISO/IEC 10589标准中。当时它仅支持CLNS网络环境，而不支持IP网络环境中的路由信息交换。再后来，IETF在RFC 1195中对IS-IS进行了修改和扩展，修改和扩展后的IS-IS称为集成IS-IS

（Integrated IS-IS）或双重IS-IS（Dual IS-IS）。集成IS-IS的制定是为了使其能够同时应用在TCP/IP网络和OSI/RM体系结构网络中，使其能够为IP网络提供动态的路由信息交换。随后由IETF扩展，使得IS-IS支持将来必将成为主流的IPv6协议。现在网络设备中的IS-IS一般都同时支持CLNP、IPv4和IPv6网络协议。

集成IS-IS是一个能够同时处理多个网络层协议（例如IP和CLNP）的路由选择协议。相反，OSPF只支持IP一种网络层协议，即OSPF仅支持IP路由，专为TCP/IP网络设计。集成IS-IS可以支持纯CLNP网络或纯IP网络，或者同时支持CLNP和IP两种网络环境，并为其提供路由功能。集成IS-IS协议经过多年的发展，已经成为一个可扩展的、功能强大、易用的IGP路由选择协议，并且在运营商网络中得到了更多的应用和部署，主要用来实现域内（也就是一个自治系统内）的IP路由选择。

3.ES-IS

ES-IS的英文全称为End System to Intermediate System Routing Exchange Protocol，翻译成中文就是终端系统到中间系统的路由交换协议，是由ISO开发，用来允许终端系统和中间系统进行配置和路由信息的交换，以推动ISO网络环境下网络层的路由选择和中继功能的操作。终端系统指用户设备，中间系统指路由器。它在CLNP网络

中，就像IP网络中的ARP、ICMP一样，为终端系统与路由器间提供路由信息交换功能。

9.3.2 IS-IS路由协议基本术语

要正确理解IS-IS路由协议工作原理，首先要理解以下这些基本专业术语。

(1) IS (Intermediate System, 中间系统)

就是TCP/IP网络中的路由器。它是IS-IS协议中生成路由和传播路由信息的基本单元。在本章下文中IS和路由器具有相同的含义。

(2) ES (End System, 终端系统)

相当于TCP/IP网络中的主机系统。ES不参与IS-IS路由协议的处理，在ISO网络中使用专门的ES-IS协议定义终端系统与中间系统间的通信。

(3) RD (Routing Domain, 路由域)

指多个使用IS-IS协议的路由器所组成的范围。

(4) Area (区域)

指IS-IS路由域的细分单元，IS-IS像OSPF一样允许将整个路由域分为多个区域。

经验之谈 其实在整个动态路由中，有三个术语最容易区分，那就是自治系统（AS）、路由域（Route Domain, RD）和区域

（Area）。总体而言，传统意义的AS为一组运行同一路由协议，并被同一组织机构管理的路由器。现在AS的概念扩展了，可以是运行多个不同路由协议，但由同一组织机构管理的一组路由器。路由域与传统的AS定义类似，也是指运行同一种路由协议，并被同一组织机构管理的一组路由器。区域是为了降低路由器的负载而划分的路由域内的子域。子域内的路由器维护子域内部具体路由信息和到达路由域内其他子域的路由信息。如果从所包含的范围来比较的话可以得出如下不等式关系： $AS \geq RD \geq Area$ 。

当每个路由协议配置不同的AS时，AS与RD是相等的，当多个不同路由协议放进一个AS中，则AS大于RD。当一个路由进程只创建一个区域时，RD与Area相等，但通常是一个路由域中要划分多个区域，所以通常是RD大于Area。它们之间可能的关系可用图9-27来表示。

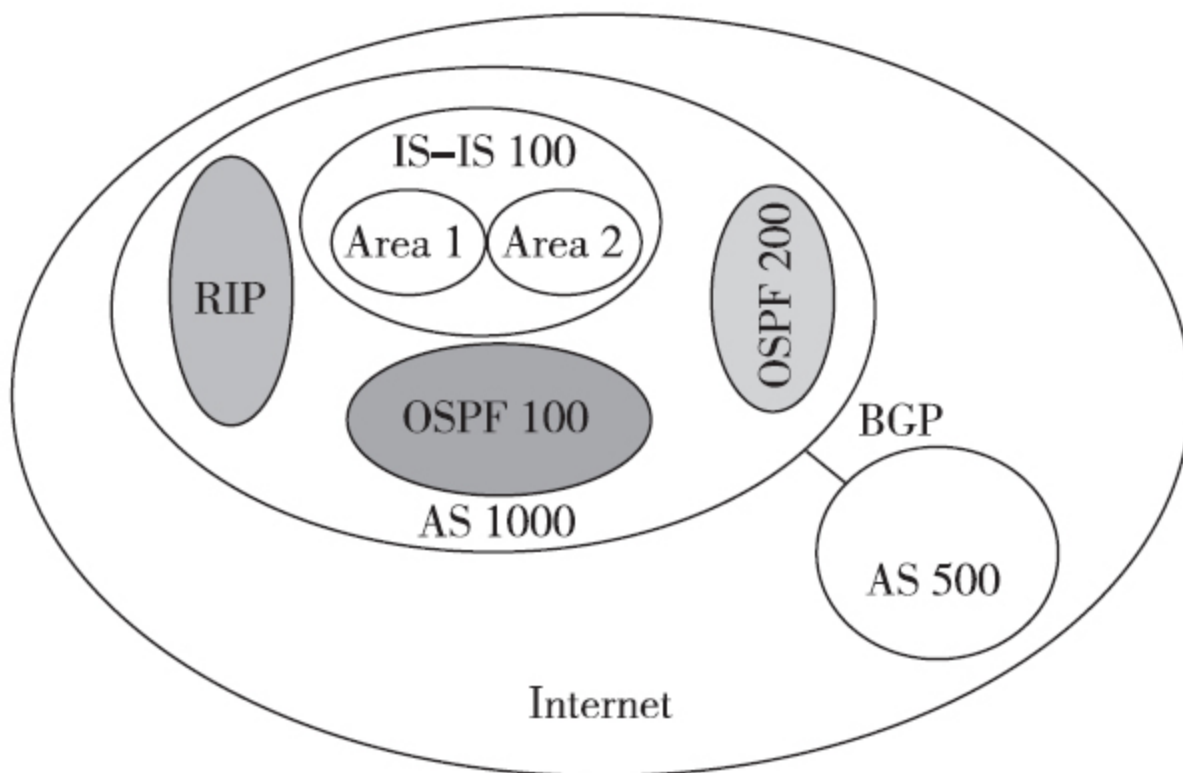


图 9-27 AS、RD与Area之间的关系

(5) LSDB (Link State DataBase, 链路状态数据库)

网络内所有链路的状态构成了LSDB，在每一个IS中都至少有一个LSDB。IS使用SPF算法（与OSPF协议使用的算法一样），利用LSDB来生成自己的路由。

(6) LSPDU (Link State Protocol Data Unit, 链路状态协议数据单元)

简称LSP。在IS-IS中，每一个IS都会生成LSP，此LSP包含了本IS的所有链路状态信息。在IS-IS路由协议中使用LSP来描述本路由器的

链路状态信息。通过LSP的泛洪，最终使整个区域内的所有中间系统拥有相同的LSDB。功能上类似于OSPF中的LSA，但是LSA并不是一种单独的报文，是封装在OSPF的协议中的。

(7) DIS (Designated IS, 指定IS)

指定IS是在广播或多路访问式网络上选举的。在IS-IS广播网络类型中，需要选举一个指定的中间系统，周期性的向其他路由器进行LSDB数据库的泛洪，功能类似于OSPF中的DR (Designated Router)。但是在OSPF中还有备份BDR (Backup Designated Router)的概念，IS-IS中没有备份的DIS的概念。

(8) NSAP (Network Service Access Point, 网络服务接入点)

即ISO网络中的网络层地址，用来标识一个抽象的网络服务访问点，描述OSI模型的网络地址结构。整个NSAP地址由两大部分组成：IDP (Inter-Domain Portion) 域间部分和DSP (Domain Service Portion) 域内服务部分。IDP类似于TCP/IP地址中的主网络ID；DSP类似于TCP/IP地址中子网络ID、主机ID和端口号。

(9) PDU (Packet Data Unit, 报文数据单元)

链路数据层传递的数据报文格式。IS-IS路由协议中又可分为Hello PDU、LS PDU、CSN PDU、PSN PDU。与OSPF协议对比，OSPF协议

承载在IP协议之上，所以PDU功能上有点类似于IP报文，而IS-IS协议是直接承载在数据链路层上，不需要经过三层协议封装。

(10) Sys ID (System ID, 系统ID)

在IS-IS路由协议中使用Sys ID唯一标识一台中间系统，必须保证在整个IS-IS路由域中System ID的唯一性。功能上类似于OSPF的Router ID。

(11) NET (Network Entity Title, 网络实体标题)

特殊的NSAP地址，其中的N-Selector部分为全0，专门为IS-IS设计。目前Cisco IOS和H3C都支持一台路由器上最多配置3个NET，方便进行网络的迁移。有关NSAP地址的具体格式将在本章后面介绍。

(12) IIH (IS to IS Hello PDU, IS到IS间的Hello PDU)

在IS-IS路由协议中使用IIH报文进行邻居的发现、建立和维护。IS-IS协议规定的Hello有三种：ESH (ES to IS Hello)、ISH (IS to ES Hello) 和IIH (IS to IS Hello)。但是纯IP的环境中只使用IIH一种Hello PDU。功能上类似于OSPF中的Hello报文。

(13) PSNP (Partial Sequence Number PDU, 部分序列号数据包)

在点到点的网络类型中用于确认链路数据信息，类似于OSPF协议中的LSAck报文；在广播网络类型中用于请求和确认链路数据信息，类似于OSPF协议中的LS Request报文和LSAck报文。

(14) CSNP (Complete Sequence Number PDU, 完全序列号数据包)

在IS-IS路由协议中用于发布完整的链路数据信息。在广播网络类型中DIS路由器生成的伪节点周期性地发送CSNP报文给其他路由器进行数据库的同步，功能上类似于OSPF协议中的DD报文。

9.3.3 IS-IS路由及路由器类型

通过前面的介绍，我们已经知道，IS-IS最初就是ISO网络的网络层路由协议，所以它的路由选择功能就与IP网络中的路由功能实现上有些不一样。下面先来简单了解一下。

1.ISO网络中的路由选择级别

在ISO网络中路由中包括了四个级别：Level-0、Level-1、Level-2和Level-3。

(1) Level-0路由

Level-0路由是指在ES与IS之间的路由，通过使用ES-IS协议进行路由信息的交换。在ES-IS协议中，ES通过侦听IS发送的IIH报文（IS到IS的Hello报文）来获知IS的存在。当ES要向其他ES发送ESH报文（ES的Hello报文）时，将同时把报文发送到IS。同样，IS也侦听ES发送的ESH报文以获知ES的存在，当有数据包要发送某个ES时，IS便根据通过ESH获取到的信息发送个特定的ES。这个过程就称为Level-0路由选择过程。

(2) Level-1路由

Level-1路由是指在同一区域内IS之间的路由。IS-IS中的区域是指在CLNP地址中拥有相同区域前缀的一组ES和IS。同一个区域中的IS之间通过交换路由信息后，便得知了本区域内的所有路径。当IS收到一个到目标地址是本区域内地址的数据包后，通过查看数据包的目的地址即可将数据包发往正确的链路或目的地。

(3) Level-2路由

Level-2路由是指在不同区域中的IS间的路由，是区域间路由。当一个IS收到的一个目的地址不是本区域CLNP地址的数据包时，便将其转发到正确的目的地，或者将数据包中继到其他区域，以便由其他区域中的IS转发到正确的目的地。

(4) Level-3路由

Level-3路由是指不同IS-IS域间的路由。**Level-3**路由类似与IP网络中的BGP（Border Gateway Protocol，边界网关协议），其目的是在不同的路由域或自治系统间交换路由信息，并将去往其他AS的数据包转发到正确的AS，以便到达最终目的地。这些AS之间可能拥有不同的路由拓扑，所以不能直接进行路由信息的交换。通常**Level-3**路由是由IRDP（Inter-Domain Routing Protocol，域间路由选择协议）来完成的，IRDP的功能类似于IP路由中的BGP路由协议。

IS-IS所完成的路由功能就是以上的Level-1和Level-2路由功能（分别简称为L1和L2级路由），也就是说，IS-IS就是用来在同一个路由域内进行区域内和区域间的路由选择，是整个ISO网络中路由选择功能的一部分。Level-1区域内的路由是通过查看地址中的系统ID后，然后选择最短的路径来完成的。Level-2区域间的路由通过查看数据包的目标区域地址（非本区域的区域地址）选择一条最短的路径来路由数据包。

2.IS-IS路由器类型

由于IS-IS只负责L1和L2级的路由器路由，这样一来，根据路由器所处的网络位置不同，又可以把这些IS-IS路由器分为三类：L1路由器（Level-1）、L2路由器（Level-2）和L1/L2（Level-1-2）路由器。无论是L1、L2，还是L1/L2路由器，都采用相同的SPF算法，分别生成各自进行数据包转发时所需的最短路径树（Shortest Path Tree，SPT）。下面分别予以介绍。

（1）L1路由器

L1路由器就是IS-IS区域内的路由器，位于一个区域内部，类似于OSPF网络中的区域内部路由器（IR）。L1路由器只与属于同一区域的L1和L1/L2路由器有直接连接关系（不能与L2路由器有直接连接关系）。L1路由器与它们交换路由信息，并维护和管理本区域内部的一

个L1 LSDB。L1 LSDB包含本区域的路由信息，到区域外的报文需转发给最近的L1/L2路由器。L1路由器只能转发区域内的数据包，或者将到达其他区域的数据包转发到距离它最近，且在同一区域的L1/L2路由器。

(2) L2路由器

L2路由器是指位于两个IS-IS区域间的路由器（一个或多个L2路由器构成骨干区域），用于连接不同非骨干区域，类似于OSPF网络中的BR（骨干路由器）。L2路由器只能与其他L2、L1/L2路由器直接连接（无论是否在同一个区域），并交换路由信息。所有L2和L1/L2路由器连接在一起即组成了骨干网，与同一区域或者其他区域的L2和L1/L2路由器形成邻居关系，维护一个L2的LSDB（该LSDB包含区域间的路由信息），负责在不同区域间通信，骨干网必须是物理连续的。IS-IS的骨干网（Backbone）指的不是一个特定的区域。骨干网在转发业务包时，L2路由器可以转发区域内的数据包，也可以转发区域间的数据包。

(3) L1/L2路由器

同时属于L1和L2的路由器称为L1/L2路由器，类似于OSPF网络中的ABR（区域边界路由器）。它既可以与同一区域的L1和L1/L2路由器形成L1级路由的邻居关系，也可以与同一区域，或者其他区域的L2

和L1/L2路由器形成L2级路由的邻居关系。L1路由器必须通过L1/L2路由器才能连接至其他区域。L1/L2路由器维护两个LSDB，L1 LSDB用于区域内路由，L2 LSDB用于区域间路由。

9.3.4 IS-IS与OSPF区域及路由器邻接关系比较

IS-IS网络与OSPF网络一样，也可以划分多个区域，但是IS-IS是工作在数据链路层的，其报文是直接以帧地格式封装；而OSPF是工作应用层的，其报文需要由网络IP包格式进行封装。所以它们两者划分区域的方法是不一样的。下面是这两者的主要不同方面。

(1) IS-IS可以有多个骨干区域

OSPF的设计基于骨干区域，而且只有一个骨干区域（区域号固定为0），所有的非骨干区域（通过ABR）必须直接与骨干相连。而IS-IS中可以有多个骨干区域，同样所有的非骨干区域（通过L1/L2路由器）必须直接与骨干相连。但IS-IS中的骨干区域号不固定。骨干区域中全是由L2路由器构成的，无L2路由器的区域是不能直接相连，中间必须经过L2路由器所在的骨干区域相连。

图9-28所示为一个运行IS-IS协议的典型网络结构。在这种拓扑结构中，Area 1是骨干区域（可以是其他区域号），该区域中的所有路由器均是L2路由器。另外4个区域为非骨干区域，它们都通过L1/L2路由器与L2路由器相连。

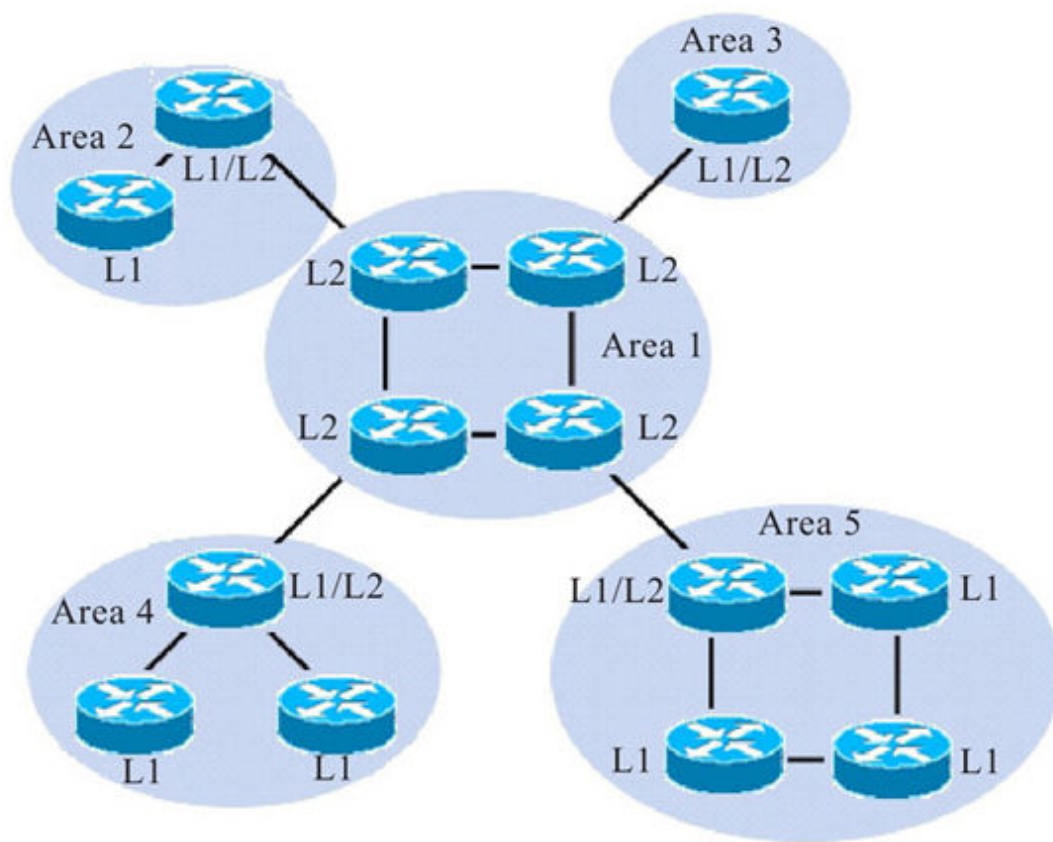


图 9-28 IS-IS网络典型拓扑结构

图9-29是IS-IS网络的另外一种拓扑结构图，其中有两个骨干区域（Area 1和Area 3）。非骨干区域中的L1/L2路由器同时与两个骨干区域的L2路由器连接，同时两个骨干区域之间也彼此连接（可以把它们看成一个大的虚拟骨干区域），共同构成了一个IS-IS骨干网。所有L2路由器和L1/L2路由器构成了IS-IS的骨干网，他们可以属于不同的区域，但必须是物理连续的。

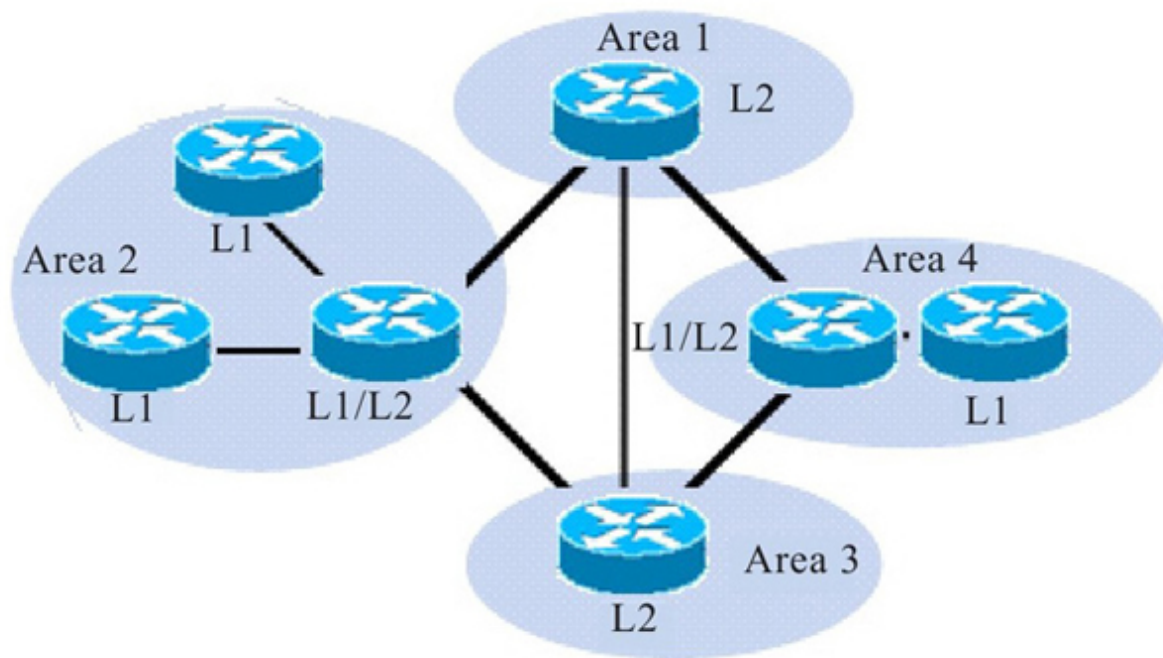


图 9-29 IS-IS网络的一种非典型拓扑结构

(2) 区域边界不同

OSPF的区域边界在ABR路由器上，OSPF的每条链路都属于某个区域。如图9-30所示，ABR的不同接口连接不同区域，且属于不同的区域，ABR不单独构成一个区域。而IS-IS的区域边界在链路上，与OSPF ABR类似的L1/L2路由器的各接口虽然也是用来连接不同区域的，但这些接口都仍属于某个非骨干区域中，链路两端分属不同区域（参见图9-28及图9-29），不同区域的边界仅体现在不同区域间连接的链路上。

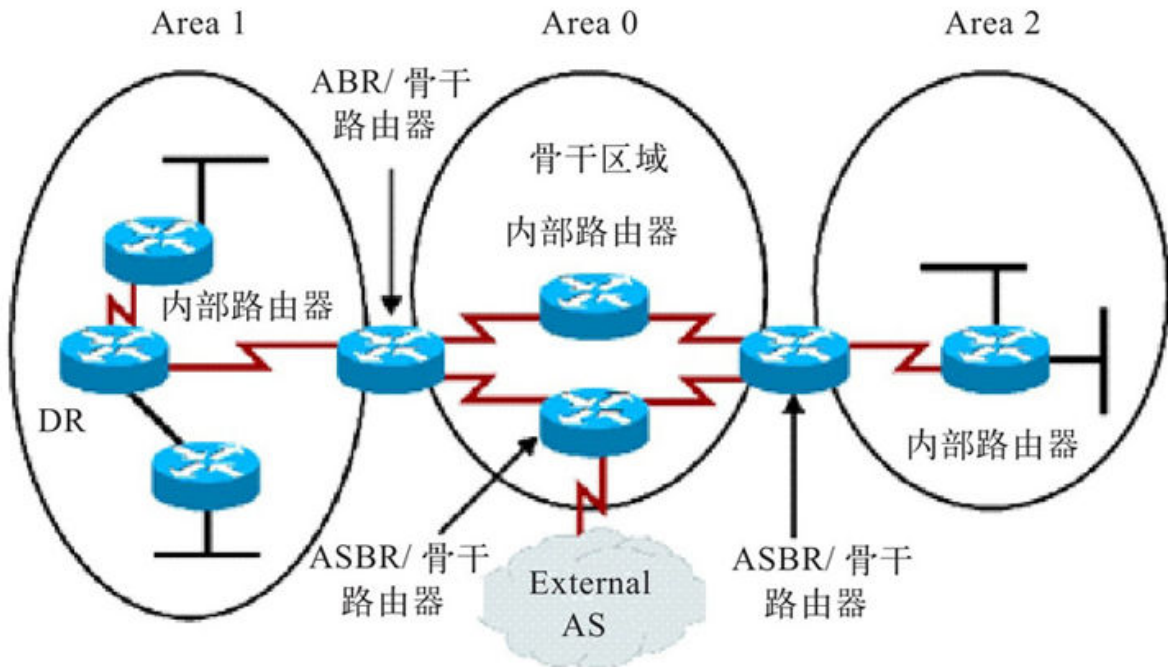


图 9-30 OSPF中的区域划分示例

(3) 不同区域间路由器的邻接关系不同

在前面我们说到，OSPF网络中DR Other仅与DR和BDR之间建立邻接关系，DR Other之间不交换任何路由信息。OSPF协议使用路由器接口来划分区域，一台路由器可能同时属于多个区域，并可以与多个区域的路由器形成邻接关系；而IS-IS协议规定路由器整体属于某个特定的区域，Level-1路由器只能建立Level-1的邻接关系，Level-2路由器只能建立Level-2的邻接关系。

另外，当IS-IS路由协议在点到点链路中，没有两次握手机制，即一方收到对方的Hello，经过合法性检查后，邻居就直接被激活了。在广播链路中，需要进行两次握手验证，邻居才可以被激活。而OSPF协

议中无论在什么链路中均需要两次握手才能建立邻居关系，可靠性更好。两台运行IS-IS的路由器在交互协议报文实现路由功能之前也必须首先建立邻接关系。在IS-IS路由协议中，只有同一层次的相邻路由器才可能成为邻接体。建立邻接关系的规则如下：

- 同一区域的L1路由器和L1/L2路由器可以建立L1级邻接；
- 同一区域的L1路由器和L2路由器不能建立任何邻接；
- 同一区域的L1/L2路由器可以与L1/L2路由器建立L1级邻接和L2级邻接；
- 同一区域的L2路由器可以与L1/L2路由器建立L2级邻接；
- 相邻区域的L1/L2路由器与L1/L2路由器只能建立L2级邻接；
- 相邻区域的L2路由器与L1/L2路由器只能建立L2级邻接。

图9-31所示是一个IS-IS网络中各级路由器之间建立的邻接关系示例，注意不带箭头的连线是物理连接，带双箭头的才是形成的邻接关系。从图中可以看出，只有区域1中的Level-1-2路由器与相同区域1中的Level-1-2路由器之间建立了Level-1邻接和Level-2邻接，其他路由器之间都只建立了一种邻接关系。建立的原则就是以上所介绍的。

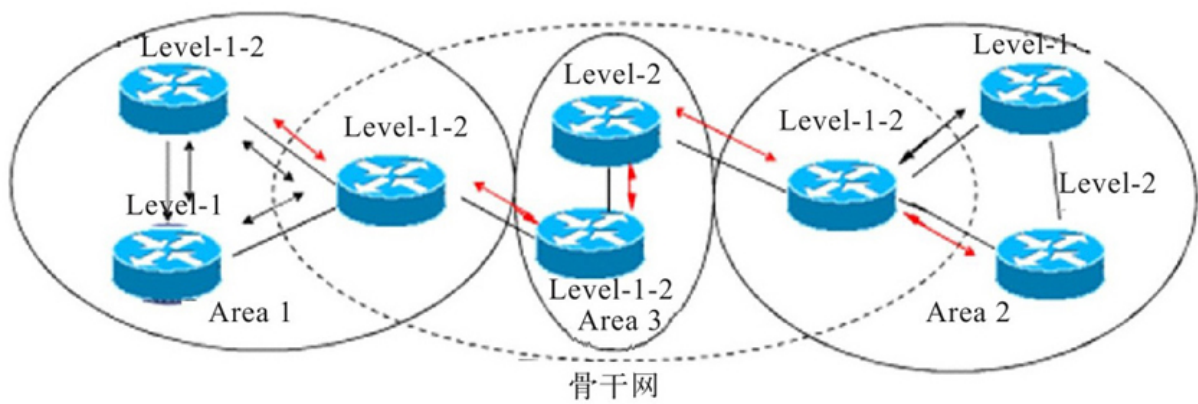


图 9-31 IS-IS路由器邻接关系示例

9.3.5 IS-IS PDU报头格式

IS-IS路由协议和其他路由协议不同，它直接运行在数据链路层之上。对等路由器间是通过传递PDU（Protocol Data Unit，协议数据单元）来传递链路信息，完成链路数据库同步的。IS-IS网络中使用的PDU主要有：IIH（IS-IS Hello，IS-IS Hello包）、LSP（Link-State Packet，链路状态包）和SNP（Sequence Number Packet，序列号包）这三种。每一种都分Level-1和Level-2两种。

以上这些PDU都包括一个报文头和可变长字段部分。报文头又包括通用报头和专用报头，对所有PDU来说，通用报头是相同的，专用报头根据PDU类型不同而不同。总体的IS-IS PDU结构如图9-32所示。



图 9-32 IS-IS PDU基本结构

IS-IS PDU通用报头格式如图9-33所示。

				字节数
Intradomain routing protocol discriminator				1
Length indicator				1
Version/Protocol ID extension				1
ID length				1
R	R	R	PDU type	1
Version				1
Peserved				1
Maximum area address				1

图 9-33 IS-IS PDU通用报头格式

IS-IS PDU通用报头格式中各字段说明如下：

□Intradomain routing protocol discriminator： 域内路由选择协议鉴别符， 占1字节， 用于标识网络层PDU的类型， 固定为0x83。

□Length indicator： 长度标识符， 占1字节， 用于标识报头部分（包括通用报头和专用报头两部分）长度， 以字节为单位。

□Version/Protocol ID extension： 版本/协议ID扩展， 占1字节， 当前值固定为0x01。

□ID length: ID长度, 占1字节, 用于标识NSAP和NET地址长度。

□PDU type: PDU类型, 占5位, 用于标识IS-IS PDU数据包的类型。值为15表示L1LAN IIH; 值为16表示L2 LAN IIH; 值为18表示L1 LSP; 值为20表示L2 LSP; 值为24表示L1 CSNP; 值为25表示L2 CSNP; 值为26表示L1 PSNP; 值为27表示L2 CSNP。

□Version: 版本, 占1字节, 当前值为0x01。

□Reserved: 保留位, 占1字节, 当前值固定为0。

□Maximum Area Addresses: 最多区域地址, 占1字节, 标识支持的最大区域数。

9.3.6 IIH PDU包格式

IS-IS Hello数据包用来建立和维持IS-IS路由器之间的邻接关系。IIH数据包包括IS-IS PDU通用报头、IIH专用报头和可变字段三部分。在IIH专用报头部分包括了发送者的系统ID、分配的区域地址和发送路由器已知的链路上邻居标识。另外，IIH有以下三种类型：

□Level-1 LAN IS-IS Hello PDU（类型号为15）：广播网中L1路由器发送的IIH。

□Level-2 LAN IS-IS Hello PDU（类型号为16）：广播网中L2路由器发送的IIH。

□点到点IS-IS Hello PDU（类型号为17）：在点对点网络上路由器发送的IIH。

前面两种统称为广播LAN IIH，后面一种称为P2P IIH。这些不同类型的IIH的PDU包格式不完全一样，图9-34所示的是广播网中L1和L2路由器发送的IIH PDU包的格式，而图9-35所示的是点对点网络中路由器发送的IIH PDU包的格式。下面介绍这些IIH PDU包专用报头部分各个字段的含义。

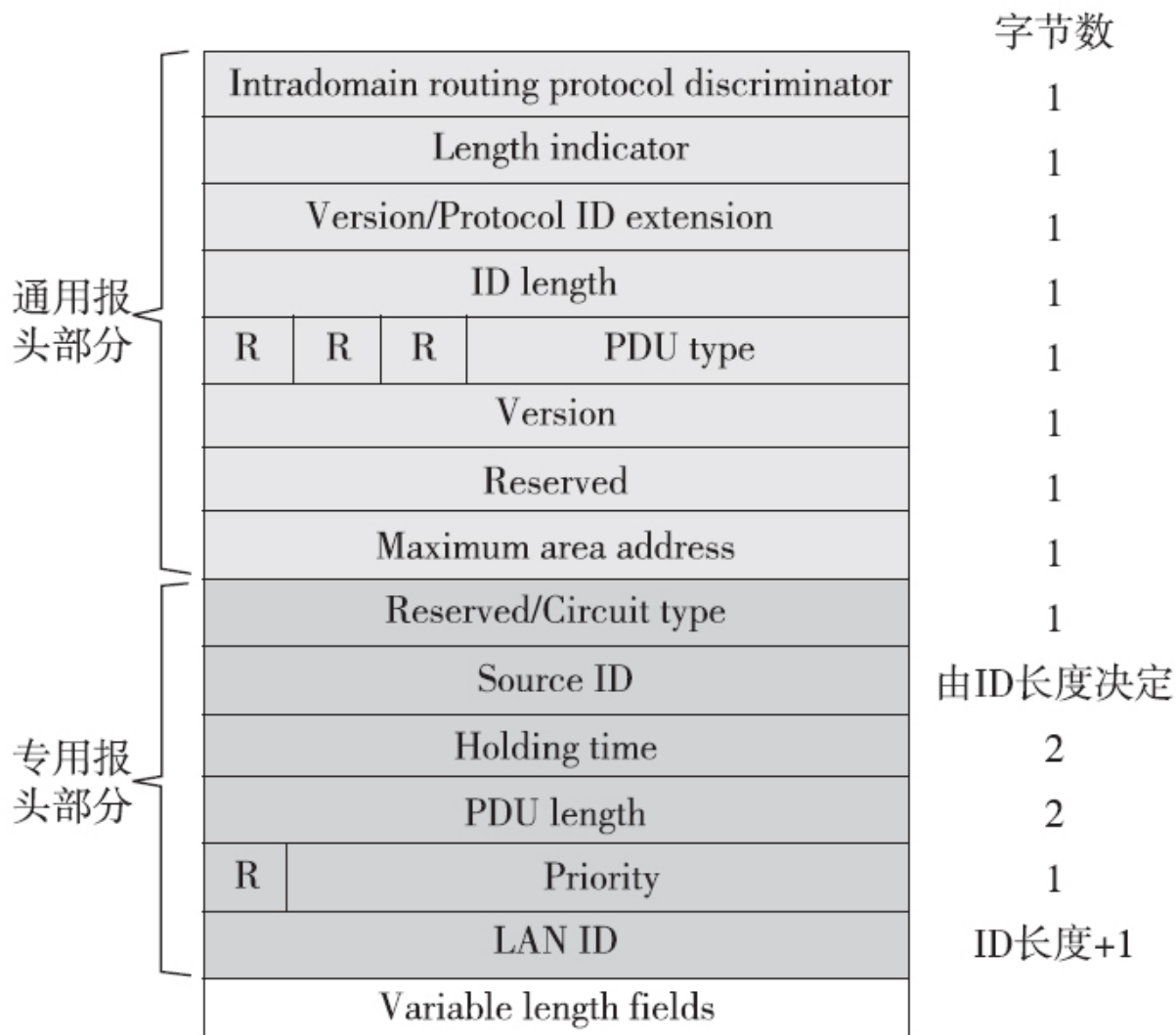


图 9-34 广播网中L1和L2路由器发送的IIH PDU包格式

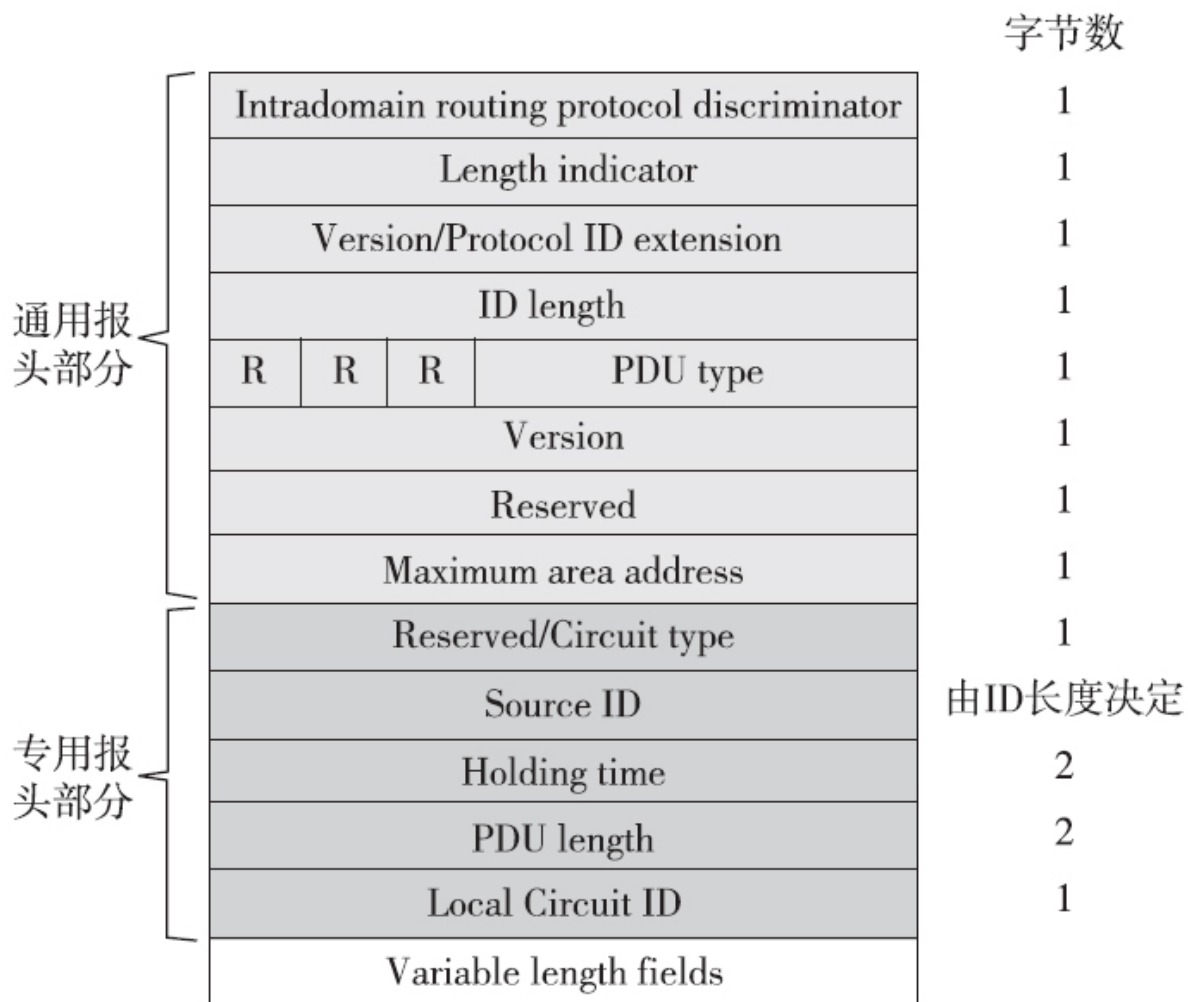


图 9-35 点对点网络IIH PDU包格式

□Reserved: 保留字段，占6位。当前没有使用，始终为0。

□Circuit type: 电路类型字段，占2位。0x01表示L1路由器，0x10表示L2路由器，0x11表示L1/2路由器。

□Source ID: 源ID字段，占1字节，标识发送该IIH PDU包的源路由器SysID（系统ID）。

□Holding time: 保持时间，占2字节，用来通知它的邻居路由器在认为这台路由器失效之前应该等待的时间。如果在保持时间内收到邻居发送的Hello PDU，将认为邻居依然处于存活状态。这个保持时间就相当于OSPF中的dead interval（死亡间隔）。在IS-IS中，默认情况下保持时间是发送Hello PDU间隔的3倍，但是这保持时间是通过指定一个Hello报文乘数（hello-multiplier）进行配置的。例如，如果Hello PDU的间隔为10s，Hello报文乘数为3，那么保持时间就是30s（10s×3）。

□PDU length: PDU长度字段，占2个字节，标识整个PDU报文的长度，以字节为单位。

□Priority: 优先级字段，占7位，标识本路由器在DIS选举中的优先级。值越大，优先级越高，路由器成为DIS的可能性越大。

□LAN ID: 局域网ID字段，占“ID长度+1”个字节，由路由器的系统ID+1个字节的伪节点ID组成，用来区分同一台DIS上的不同LAN。

对比图9-34和图9-35可以看出，点对点网络IIH PDU与广播网络中的IIH PDU包格式基本一样，只是没有了广播网络中的Priority和LAN ID这两个字段，另外新增了一个Local Circuit ID字段，用来标识本地链路ID。

9.3.7 LSP PDU包格式

与OSPF一样，运行IS-IS路由选择协议的路由器也是通过收集其他路由器泛洪的链路状态信息来构建自己的链路状态数据库的。在OSPF中，OSPF路由器通告链路状态信息是通过LSA实现的。在IS-IS中，与LSA具有同样功能的包含链路状态信息的报文称为LSP。LSP用来承载和泛洪路由器的链路状态信息，并且LSP是路由器进行SPF计算的依据，包含了由IS-IS路由器产生的描述其周围环境的路由选择信息。LSP共分为两种类型：

(1) Level-1 LSP（类型号为18）

Level-1 LSP是由支持Level-1的路由器产生的，会在Level-1区域中泛洪。Level-1 LSP集由所有在Level-1 LSPDB（LSP Database，LSP数据库）区域中的Level-1路由器的产生组成。区域中的所有Level-1路由器将有相同的Level-1 LSPDB和相同的区域网络连接映射。

(2) Level-2 LSP（类型号为20）

Level-2 LSP是由支持Level-2的路由器产生的，通过Level-2子域泛洪。Level-2 LSP集由所有在Level-2 LSPDB域中的Level-2 IS的产生组成。所有Level-2路由器将有相同的Level-2 LSPDB和相同的Level-2子域连接映射。

以上两种LSP PDU具有相同的包格式，如图9-36所示。下面是LSP专用报头部分各字段介绍。

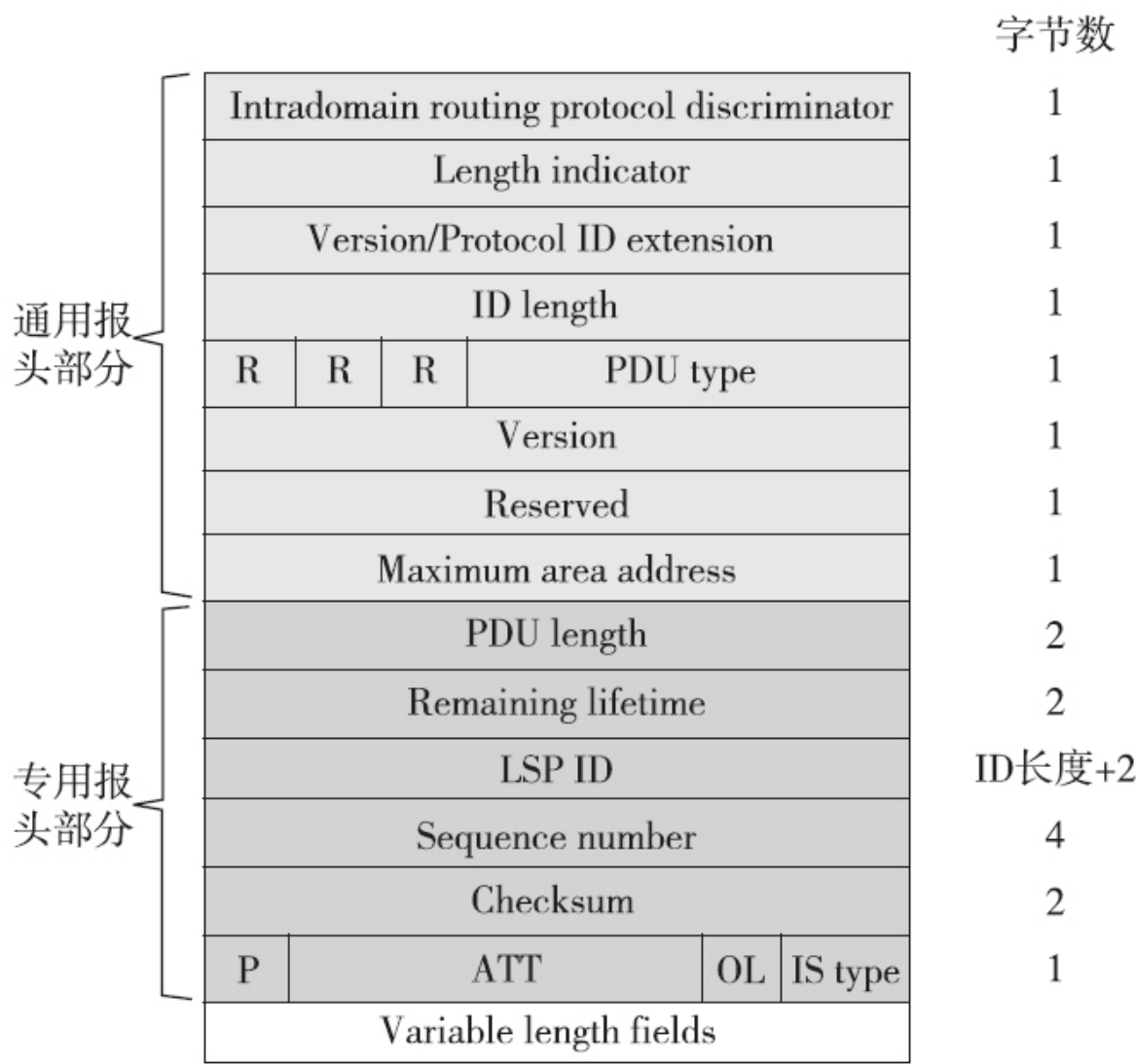


图 9-36 LSP PDU包格式

□PDU length: PDU长度字段，占2字节，标识整个PDU报文的长度。

□Remaining Lifetime: 剩余生存时间字段，占2字节，标识此LSP到期前的生存时间，以秒为单位。当生存时间为0时，LSP将被从链路状态数据库中清除。

□LSP ID: LSP标识符字段，占“系统ID长度+2”个字节，用来标识不同的LSP和生成LSP的源路由器。LSP ID包括SysID、伪节点标识符（Pseudonode ID）和LSP编号三个部分。

□Sequence number: 序列号字段，占4字节，标识每个LSP包的序列号。

□Checksum: 校验和字段，占2字节，用于接收端校验所传送LSP的完整性和正确性。当一台路由器收到一个LSP，在将该LSP放入到本地链路数据库和将其再泛洪给其他邻接路由器之前，会重新计算LSP的校验和，如果校验和与LSP中携带的校验和不一致，则说明此LSP传输过程中已经被破坏。

□P: 分区字段，占1位，表示区域划分或者分段区域的修复位，仅与Level-2 LSP有关。当P位被设置为1时，表明始发路由器支持自动修复区域的分段情况。

□ATT (Attached): 区域关联字段，占4位，表示产生此LSP的路由器与多个区域相连。虽然ATT位同时在L1 LSP和L2 LSP中进行了定

义，但是它只会在L1 LSP中被置位，并且只有L1/2路由器会设置这个字段。

□OL: 超载字段，占1位，表示本路由器因内在不足而导致LSDB不完整。被设置了超载位的LSP不会在网络中进行泛洪，并且当其他路由器收到设置了超载位的LSP后，在计算路径信息时不会考虑此LSP，因此最终计算出来的到达目的地的路径将绕过超载的路由器。设置超载位还可以使数据的传输路径绕过某个特定的路由器。

□IS type: 路由器类型字段，占2位，表示了此LSP是来自L1路由器还是L2路由器，也表示了收到此LSP的路由器将把这个LSP放到L1链路状态数据库还是L2链路状态数据库。0x01表示L1，0x10表示L2。

9.3.8 SNP PDU包格式

SNP（Sequence Number PDU，序列号PDU）包含一个或者多个LSP的汇总描述，用于确认邻居之间最新接收的LSP，作用类似于确认报文。

SNP可分为CSNP（Complete Sequence Number PDU，完整SNP）和PSNP（Partial Sequence Number PDU，部分SNP）。CSNP包括LSDB中所有LSP的摘要信息，从而可以在相邻路由器间保持LSDB同步。在广播网上，CSNP由DIS定期发送（默认发送周期为10秒）；在点对点网络中，CSNP只在第一次建立邻接关系时发送。PSNP仅用来列举最近收到的一个或多个LSP的序号，能够一次对多个LSP进行确认。当LSDB不同步时，也用PSNP请求邻居发送新的LSP。

CSNP PDU包格式如图9-37所示，PSNP PDU包格式如图9-38所示。下面对这两种SNP PDU包中专用报头部分的各字段具体介绍。

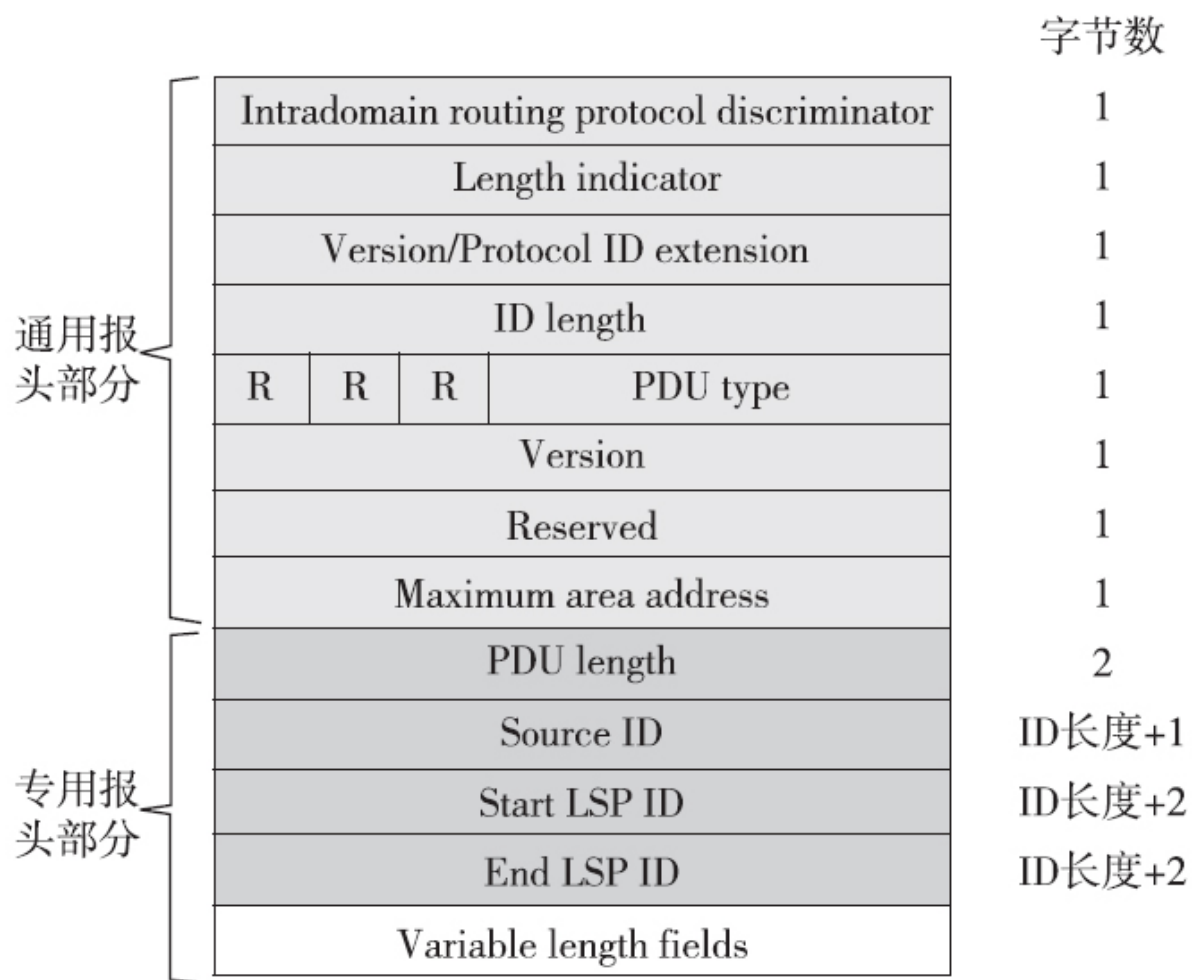


图 9-37 CSNP PDU包格式

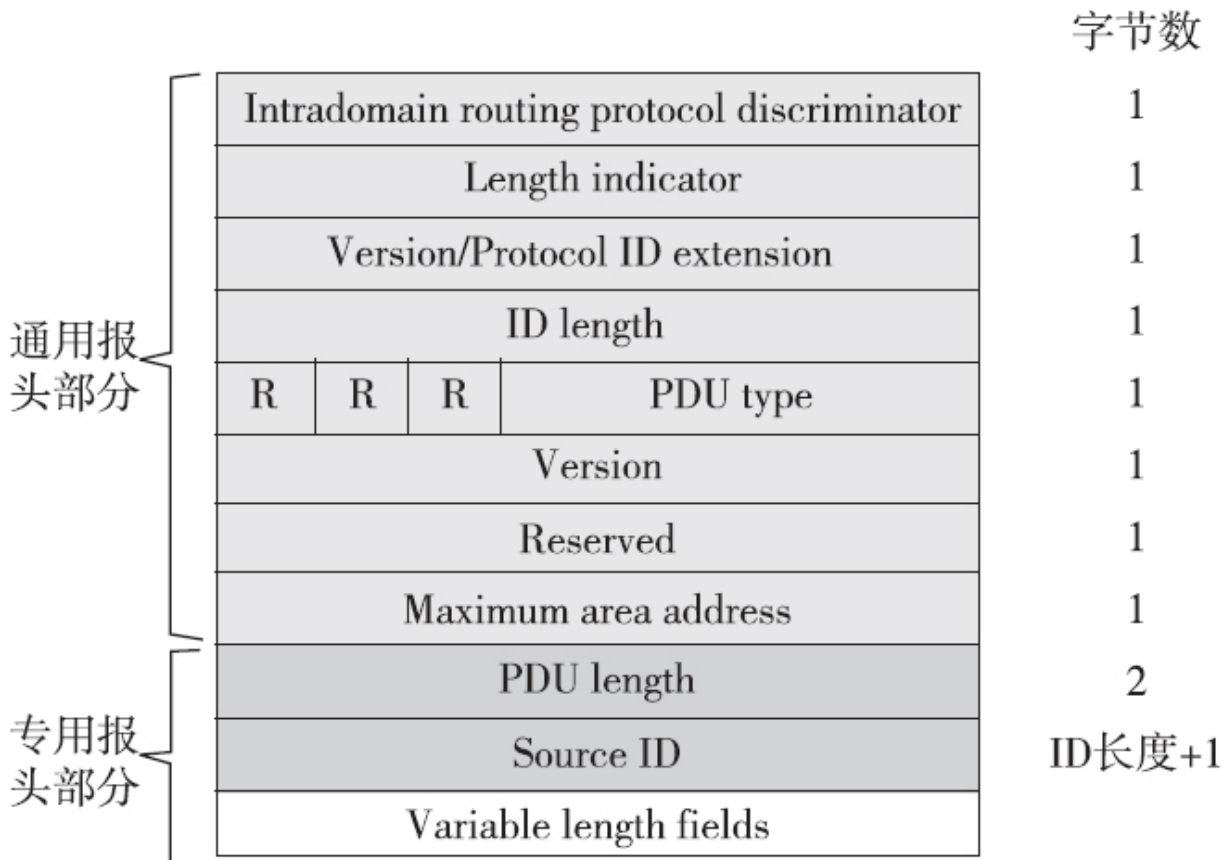


图 9-38 PSNP PDU包格式

□PDU length: PDU长度字段，占2字节，标识整个PDU报文的长度。

□Source ID: 源ID字段，占“系统ID长度+1”个字节，标识发送该CSNP PDU的路由器的SysID。

□Start LSP ID: 起始LSP ID字段，占“系统ID长度+2”个字节，表示在下面的可变字段中描述的LSP范围中的第一个LSP ID号。

□End LSP ID: 结束LSP ID字段，占“系统ID长度+2”个字节，表示在下面的可变字段中描述的LSP范围中的最后一个LSP ID号。

9.3.9 IS-IS PDU可变字段格式

大家或许从上面介绍的各种PDU包格式已经看到，除了报头（包括通用报头和专用报头）部分，最后还有一个部分，就是“可变长字段”（Variable length fields）部分。这个可变成字段部分就是各种PDU包真正内容部分，是整个PDU包的核心部分。因为这部分内容都是以“Type-Length-Value”，即“类型-长度-值”格式列出的，所以又称为TLV部分。

TLV编址方式由三大部分组成：T（Type），即PDU包类型，不同类型由不同的值定义；L（Length），即Value字段的长度，以字节为单位；V（Value），即包的真正内容，是最重要的部分。但在ISO 10589和RFC 1195这两个当前IS-IS标准中，使用“code”（代码）替换了前面所说的“Type”部分，所以这种报文编址方式通常又称CLV（Code-Length-Value），如图9-39所示。

字节数	
Code	1
Length	1
Value	=Length

图 9-39 可变长字段部分格式

图9-39所示三个字段的说明如下：

□Code: 代码字段，占1字节，表示PDU类型，不同的IS-IS PDU使用不同的类型。

□Length: 长度字段，占1字节，表示Value字段的长度，最大值为255字节。

□Value: 值字段，长度是可变的，表示实际承载的PDU内容，最大为255字节。

在IS-IS PDU所使用的各种TLV中，既有ISO 10589中定义的，也有RFC 1195中定义的。ISO中定义的TLV用于CLNP网络环境，但是其中的大多数也用于IP网络环境。RFC中定义的TLV只用于IP环境。也就是说，对于一个IS-IS PDU，后面既可以携带支持CLNP协议的TLV，又可以携带支持IP协议的TLV。如果一个路由器不能识别一个TLV，那么将忽略它。

至于IS-IS的各种TVL，在此不作具体介绍，大家可参考相关资料。

9.3.10 IS-IS的两种地址格式

IS-IS中包括两种地址，一是用来标识网络层服务的NSAP（Network Service Access Point，网络服务访问点）地址，二是用来标识设备的NET（Network Entity Title，网络实体标题）地址。下面分别予以介绍。

1.NSAP地址格式

NSAP地址用来标识网络层服务，每种服务对应一个NSAP地址。NSAP地址由两个主要部分组成，即IDP（Inter-Domain Portion，域间部分）和DSP（Domain Service Portion，域服务部分），如图9-40所示。与IP地址中由网络ID和主机ID两部分组成类似，IS-IS NSAP地址中的IDP部分相当于TCP/IP网络IP地址中的主网络ID部分，而DSP部分则相当于TCP/IP网络IP地址中的子网ID、主机ID和端口号总和。

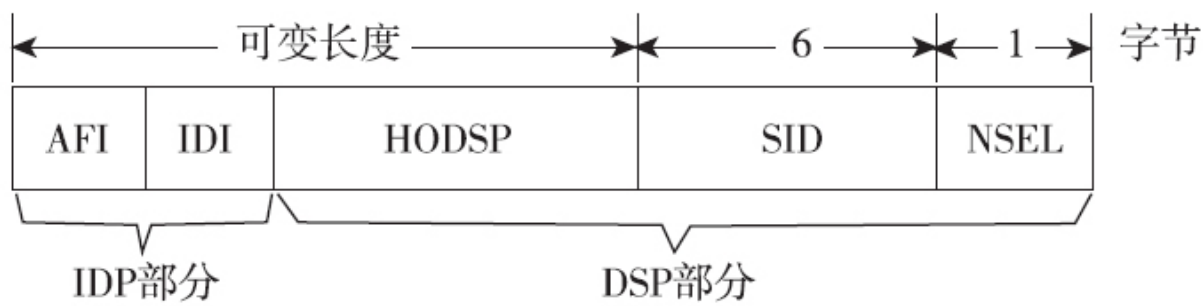


图 9-40 IS-IS NSAP地址格式

IDP又由以下这两个子部分组成:

□AFI (Authority and format ID, 机构与格式ID): 用来标识地址格式和对应地址的分配机构, 占1字节。AFI等于49的地址是私有地址, 就像IP地址中的局域网地址一样, 而AFI等于39或47的地址属于ISO注册地址, 相当于IP地址的公网地址。

□IDI (Inter-Domain ID, 域间ID): 用来标识、区分AFI字段下不同的域, 最长可达10字节。

DSP由以下三个子部分组成:

□HODSP (High Order DSP, DSP高位), 用来在一个域中分割多个区域, 相当于IP地址中的子网ID部分。

□SID (System ID, 系统ID), 占6字节, 用来区分主机, 通常以MAC地址进行标识, 相当于IP地址中的主机ID部分。当IS工作在Level-1, 则在所有同区域中的Level-1路由器的系统ID必须唯一; 当IS工作在Level-2, 则在同一个路由域中所有路由器的系统ID必须唯一。

□NSEL (Network-Selector, 网络选择器), 占1字节, 用来指示选定的服务, 相当于TCP协议中的端口号。在IS-IS路由选择过程中, 没有使用NSEL, 所以NSEL始终保持为00。

从以上可以看出，NSAP地址中包含了很多不同的字段，看起来有些复杂。但实际上可以将NSAP地址进行简化，其中各种字段可以归类为三个部分：区域地址、SysID和NSEL，如图9-41所示。把AFI、IDI和HODSP这三个字段合并为“Area Address”（区域地址）字段，并且规定整个NSAP地址最长为20字节，由于SID为6字节，NSEL为1字节，那么“区域地址”部分可为1~13字节。由于一般情况下1字节足够用于定义区域ID，所以在大多数的IS-IS实现中NSAP地址最小长度为8字节。对于IP应用程序而言，在NSAP地址中，1字节定义AFI，最少2字节定义实际的区域信息（IDI），6字节定义SID，1字节定义NSEL，所以此种情况下的NSAP地址最少为10字节。

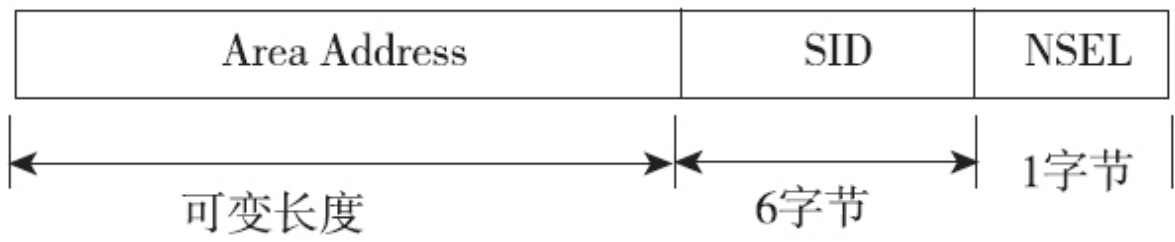


图 9-41 简化的NSAP地址格式

2.NET地址格式

在IS-IS路由选择过程中，没有使用NSAP地址中的NSEL部分，所以NSEL始终保持为00。当NSEL为00时，我们就称这个NSAP地址为NET地址。NET地址用来标识网络层实体或过程，而不是服务。

NET地址用来唯一地表示IS-IS路由选择域中的OSI主机，路由器使用NET地址来标识自己。路由器在发送的链路状态数据包（LSP）中用NET来标识自己，这类似于OSPF发送的LSA中的路由器ID（Router ID）。通常情况下，一台路由器配置一个NET即可，当区域需要重新划分时，由于区域地址最多可配置三个，所以NET最多也只能配三个。

整个NET地址分成三个部分：区域地址（Area Address）、系统ID（System ID）和NSEL（网络选择器）。

如果一NSAP地址为49.0001.aaaa.bbbb.cccc.00，根据图9-41可以很快得出，代表的“区域地址”为49.0001（因AFI=49，所以它是一个私有地址），系统ID为aaaa.bbbb.cccc，NSEL为00。如果另一个NSAP地址为39.0f01.0002.0000.0c00.1111.00，则可以得出“区域地址”为39.0f01.0002（因AFI=39，所以它是一个公有地址），系统ID为0000.0c00.1111，NSEL为00。

9.3.11 IS-IS与OSPF的比较

IS-IS与OSPF是最为流行的IGP路由协议，它们存在许多相同或者相似之处，如它们都是链路状态路由协议，都使用了相同的最短路径优先（SPF）路由算法，都可划分区域等。本节对这两个路由协议做一个综合比较。具体的比较如表9-5所示。

表 9-5 IS-IS 与 OSPF 的比较

相同点	
IS-IS 和 OSPF 是链路状态路由协议的两个最典型的代表，都采用 SPF 算法来计算路由	
由于具有快速收敛、无环路等特点，IS-IS 和 OSPF 都能很好地支持大型网络，但从全球的部署来看，采用 OSPF 的还是占了多数，而 IS-IS 在近几年开始得到比较多的应用	
IS-IS 和 OSPF 一样采用 Hello 协议来维护邻居关系，但 IS-IS 的 Hello 协议与 OSPF 具体实现上有所不同	
为了控制链路状态数据库的规模和复杂度，IS-IS 和 OSPF 在广播网络上都选举 DR 来担任数据库同步的主要角色，但在细节处理上还是有较大差别的	
IS-IS 和 OSPF 对路由开销的度量（metric）都采用了接口可配置的 cost，能够比较正确地反映网络的实际情况	
IS-IS 和 OSPF 都采用分层路由的概念，都有骨干区域，为网络规划提供了比较灵活而且实际的设计方案	
不同点	
IS-IS	OSPF
IS-IS 可以支持 CLNP 和 IP 两种网络环境	OSPF 仅支持 IP 网络环境
IS-IS 所使用的数据包被直接封装到数据链路层帧中	OSPF 数据包被封装在 IP 报文中
IS-IS 是 ISO CLNS 中的一个网络层协议	OSPF 不是网络层协议，它运行在 IP 之上
IS-IS 使用 LSP 承载所有的路由选择信息	OSPF 使用不同类型的 LSA 承载路由选择信息
IS-IS 利用 TLV 可以灵活的对协议进行扩展	OSPF 很难进行扩展
IS-IS 可以忽略不支持的 TLV	网络中所有路由器都必须能够识别所有 LSA
IS-IS PDU 可以承载多个 TLV 字段，只有一个报头，节省带宽	1 类、2 类 LSA 可以承载多个 IP 前缀；3 类、4 类、5 类 LSA 只能承载单个 IP 前缀，如果需要发送多个 IP 前缀信息，需要多个 LSA
IS-IS 仅支持广播类型链路与点到点类型链路	OSPF 可以支持多种网络类型：广播、点到点、NBMA、点到多点和按需电路（Demand Circuit）
IS-IS 邻接关系建立过程简单，仅 3 步	OSPF 需要通过多种状态建立邻接关系
数据库同步在建立邻接关系之后	数据库同步在邻接关系建立之前
IS-IS 路由器只属于一个区域，基于节点分配区域	OSPF 路由器可以属于多个区域，典型的是 ABR，OSPF 基于接口分配区域

(续)

不同点	
IS-IS	OSPF
IS-IS 的区域边界在链路上	OSPF 的区域边界在路由器上
IS-IS 的 Level-1 区域（非骨干区域）为末节（stub）区域，除非使用路由渗透（Route Leaking）机制	默认情况下，OSPF 非骨干区域不是 stub 区域，但可以配置为 stub 区域
IS-IS 仅在点到点链路上的扩散是可靠的，在广播链路中通过 DIS 周期性的发送 CSNP 来实现可靠性	OSPF 在所有链路上的扩散都是可靠的
IS-IS 中没有备份 DIS	OSPF 中要选举 BDR，以接替 DR 的角色
IS-IS 中的 DIS 可以被抢占	OSPF 中的 DR 不能被抢占
DIS 以 3 倍的频率发送 Hello PDU	DR 以正常的 频率发送 Hello 报文
默认情况下，IS-IS 的 LSP 最大生存时间为 1200s，刷新间隔为 900s，而且定时器的值可调	OSPF 的 LSA 的老化时间为 3600s，刷新间隔为 1800s，而且是固定值
默认情况下，IS-IS 的接口 cost 值为 10	默认情况下，OSPF 的接口 cost 值根据带宽进行计算
默认情况下，IS-IS 保持时间（holding-time）为 30s，而且在建立邻接关系时不需要双方的保持时间匹配	默认情况下，OSPF 的保持时间（dead-interval）为 40s，而且为了建立邻接关系，必须使双方的保持时间一致
IS-IS 通过将 Hello PDU 的大小填充至接口 MTU 大小来检查双方的 MTU 是否匹配	OSPF 通过在 DBD 报文中嵌入接口 MTU 字段来检查双方的 MTU 是否匹配

9.3.12 IS-IS最短路径计算和路由表生成原理

当路由器中的LSPDB内容发生改变时，每个路由器都使用SPF独立地重新计算最短路径。这个算法基于前面介绍的Dijkstra算法。SPF的输出是一组参数集合，如目的地址和下一跳地址。目的地地址是协议参数，例如对于IP网络来说是一个地址前缀，对于CLNP网络来说是终端系统的NSAP。在IS-IS中支持多个等价路径，此时到达相同目的地址就可能有不同的下一跳。

SPF计算是由每个路由器独立完成的，这时就会存在同时有到达同一目的地址的Level-1和Level-2两种路径，但Level-1路径优先。链路状态路由协议都是基于近邻关系的，每个路由器公开其链路的开销和状态。所以区域里每个路由器知道区域内所有活动的链路，并知道有关这些活动链路所标识路由器发起的信息，然后把该状态信息传播到其他路由器。传播信息使所有路由器保持为同一数据库。每个路由器用唯一的地址来标识，从而避免了循环。

1.IS-IS最短路径计算原理

应SPF算法在计算最短路径时，每台路由器都是以自己为根结点，其他路由器为叶子结点，根据网络拓扑信息生成一棵最短路径树（SPT），然后计算出根结点到各个目的地的最短路径。由于IS-IS路由

协议采用分层的网络结构，SPF算法根据Level-1和Level-2两个数据库分别独立计算各自的SPT树。下面以图9-42中所示的A路由器为例，介绍IS-IS路由的计算过程。

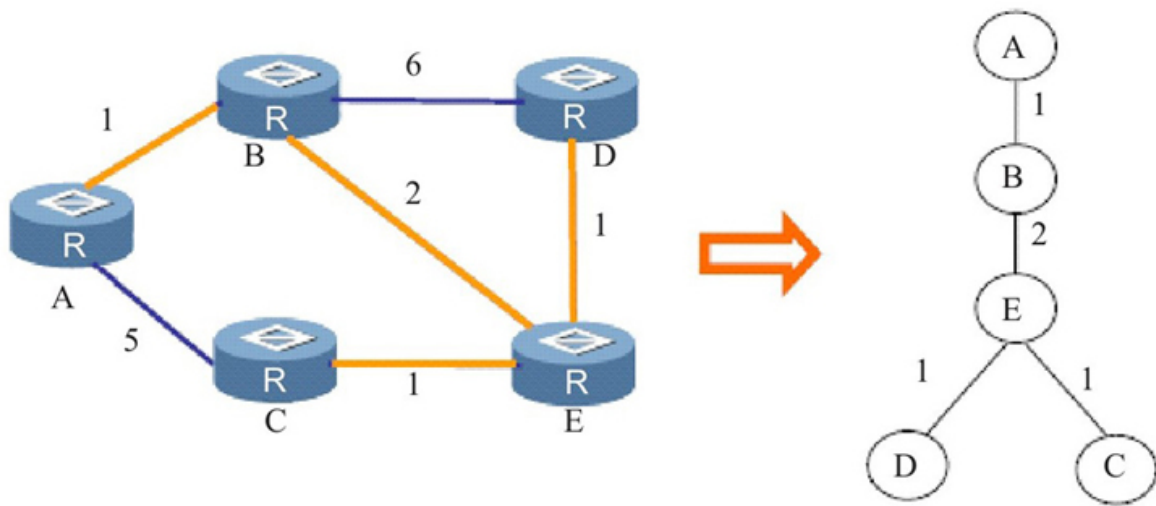


图 9-42 IS-IS最短路径计算示例

A路由器根据本地链路数据库信息，以自己为根路由器，其他路由器为叶子结点，使用SPF算法产生了图9-42所示的SPT最短路径树。然后根据最短路径树上的附带的开销值计算出根结点到各个目的网段的路由。由于产生的SPT树为单向，且不可逆转的路径树，可以从算法上保证区域内无路由环路的发生。

2.IS-IS路由表的生成原理

通过可靠的扩散算法各路由器将其他路由器扩散来的拓扑信息收集起来，组成一张一致、完整的拓扑图，依靠SPF算法来计算出自己的

路由表。IS-IS路由协议中区域中所有的路由器拥有相同的链路状态数据库LSDB，但是每台路由器都是以自己为根结点，其他路由器为叶子结点，生成了一棵单向的、不可逆转的最短路径树SPT。然后根据SPT树以及链路相应的开销值计算出各自的路由表。

图9-43显示了如何使用LSP来创建一个网络地图。假设网络拓扑结构是一个拼图，每个LSP看成一个拼图片。图中的所有路由器适用于在一个区域中的所有Level-1路由器或者一个Level-2子域中的所有Level-2路由器。

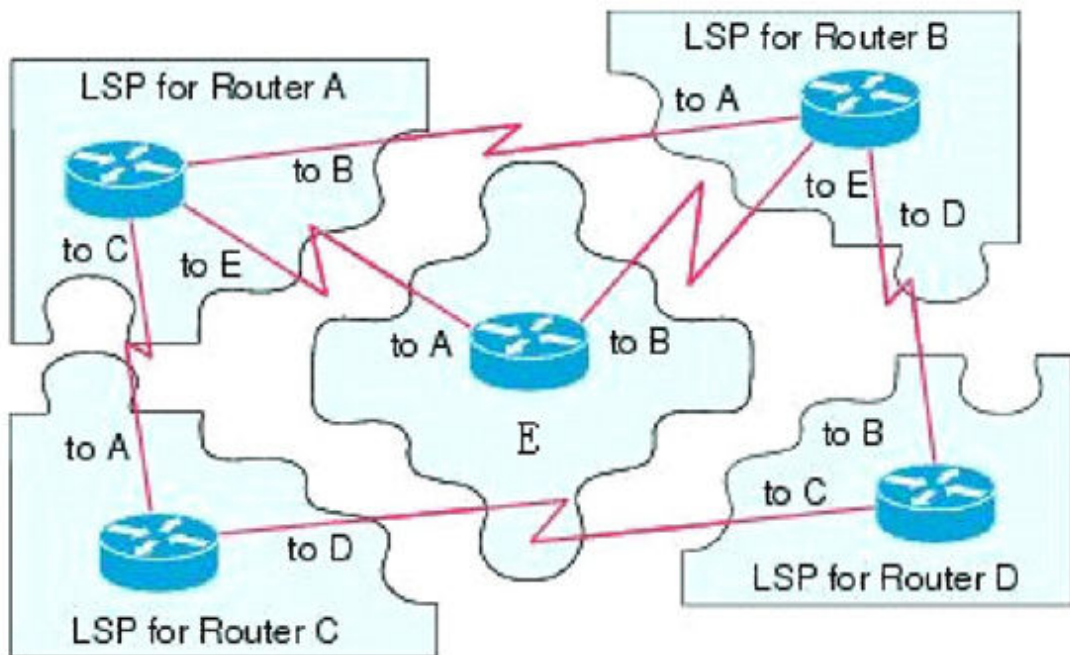


图 9-43 IS-IS网络地图示例

图9-44显示了在IS-IS网络中的这些邻居路由器之间形成邻接关系后，带有完整更新链路状态数据的每个路由器。从中可以看到各路由

器的链路状态信息是完全一致的，包括了网络中各路由器的网络连接信息，这就是LSDB同步的结果。图中的路由器适用于在一个区域中的所有Level-1路由器或者一个Level-2子域中的所有Level-2路由器。

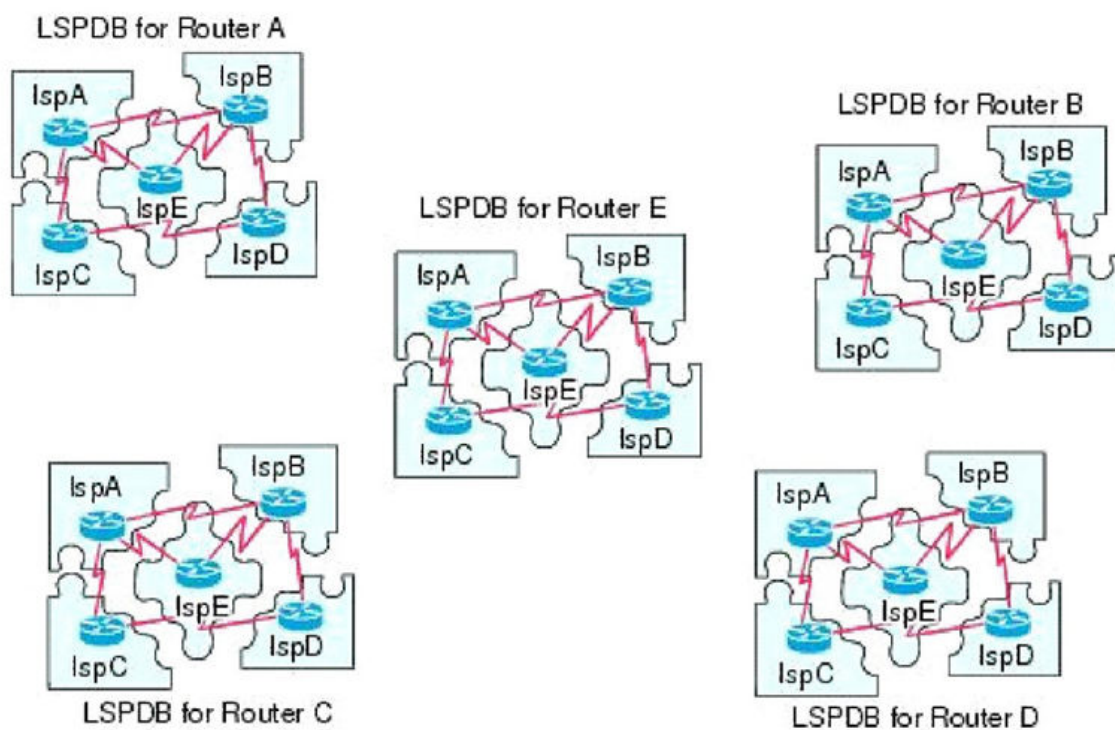


图 9-44 LSDB同步后的各路由器路由表

9.4 BGP

BGP（Border Gateway Protocol，边界网关协议）是一个在包含独立路由策略（也就是独立**AS**）的分离路由域间的无环路路由协议。很显然，相比前面介绍的**RIP**、**OSPF**这些内部网关路由协议来说，**BGP**更加复杂。也正如此，**BGP**主要用于大型网络和**ISP**运营商，在此我们仅作一般意义上的了解即可。

目前最新的版本是**BGPv4**，包括支持4字节的**AS**号和多协议扩展，以允许**BGP**为**IP**组播路由和多个三层协议地址（包括**IPv4**、**IPv6**）、**VPNv4**、**CLNS**（Connectionless Network Service，无连接网络服务）和二层**VPN**（**L2VPN**）承载路由信息。

9.4.1 BGP概述

BGP是一个设计在组织网络间（也就是路由域间，或者说是路由域边界）提供无环路的域间路由协议。它使用可靠传输协议**TCP 179**号端口（这是目的端口，本地端口可以随意）来传输**BGP**通信，因为**TCP**是一个面向连接的协议。

BGP主要用于连接一个本地网络到外部网络，以访问**Internet**或者与其他公司网络。当用于连接外部其他公司网络时，将创建一个对等

的外部BGP（external BGP，eBGP）会话。虽然BGP是一个外部网关（Exterior Gateway Protocol，EGP）协议，但许多公司内部网络也正变得越来越复杂，BGP也可以用于在一个公司内部网络间的互连。在同一公司内部BGP peer（可理解为BGP会话两端的路由器）之间通过内部BGP（internal BGP，iBGP）对等会话交换路由协议。

BGP使用的路径矢量（path-vector）路由算法，是由距离矢量（distance-vector）路由算法和AS路径（AS-path）环路检测组成的。通过路径矢量算法可与其他BGP speaking（可理解为分支）连网设备在进行路由更新时交换网络可达信息。网络可达信息包括网络号、路径参数属性和一个路由传输到达目的地必须通过的AS列表。这个AS列表包含在AS路径属性中。BGP通过拒绝任何包含本地AS号的路由更新来实现无路由环路的，因为包含本地AS号的路由更新会表示路由已通过这个AS进行了传输，这样就形成了环路。

BGP选择一条单一路径，默认为到达目的网络或主机的最佳路径。这个最佳路径选择算法是通过分析路径属性来确定哪个路径将被作为最佳路径安装在BGP路由表中。每个路径携带用于BGP最佳路径分析的公认强制（Well-known mandatory）、公认自由选择（Well-known discretionary）、可选传递（Optional transitive）和可选非传递（Optional non-transitive）属性。

总体而言，BGP路由协议具有以下基本特性：

□是一种外部网关协议（Exterior Gateway Protocol, EGP），与OSPF、RIP等内部网关协议（Interior Gateway Protocol, IGP）不同，其着眼点不在于发现和计算路由，而在于控制路由的传播和选择最佳路由。

□使用TCP作为其传输层协议（端口号179），提高了协议的可靠性。

□支持CIDR（Classless Inter-Domain Routing，无类别域间路由）。

□路由更新时，BGP只发送更新的路由，大大减少了BGP传播路由所占用的带宽，适用于在Internet上传播大量的路由信息。

□BGP路由通过携带AS路径信息彻底解决路由环路问题。

□提供了丰富的路由策略，能够对路由实现灵活的过滤和选择。

□易于扩展，能够适应网络新的发展。

9.4.2 BGP AS

前面介绍了OSPF、IS-IS路由协议中的AS（Autonomous System，自治系统）这个概念，在BGP协议中同样有AS的概念。BGP的AS用于划分整个外部网络为一个个应用本地路由策略的路由子域，这样公司通过BGP可以简化路由域管理和统一策略配置。统一策略配置对于BGP有效处理到目的网络的路由非常重要。

在BGP的每个路由子域中，可以支持多个不同的路由协议。但是，每个路由协议是独立管理的，其他路由协议通过重新发布（redistribution）功能可以与BGP动态交换路由信息。分隔的BGP AS动态通过eBGP对等会话交换路由信息；同一个AS中的BGP对端通过iBGP对等会话交换路由信息。

图9-45举例说明了在分隔AS中的两个路由器可以通过BGP协议进行连接。Router A和Router B是两个使用公用AS号（也就是这些AS号不是保留和私用的，可以应用于公共网络，类似于公网IP地址）的独立路由子域上的ISP路由器。这两个路由器通过Internet来传输通信。Router A和Router B是通过eBGP对等会话进行连接的。每个直接连接Internet的公共AS各自分配一个唯一的AS号，用于标识BGP进程和AS。

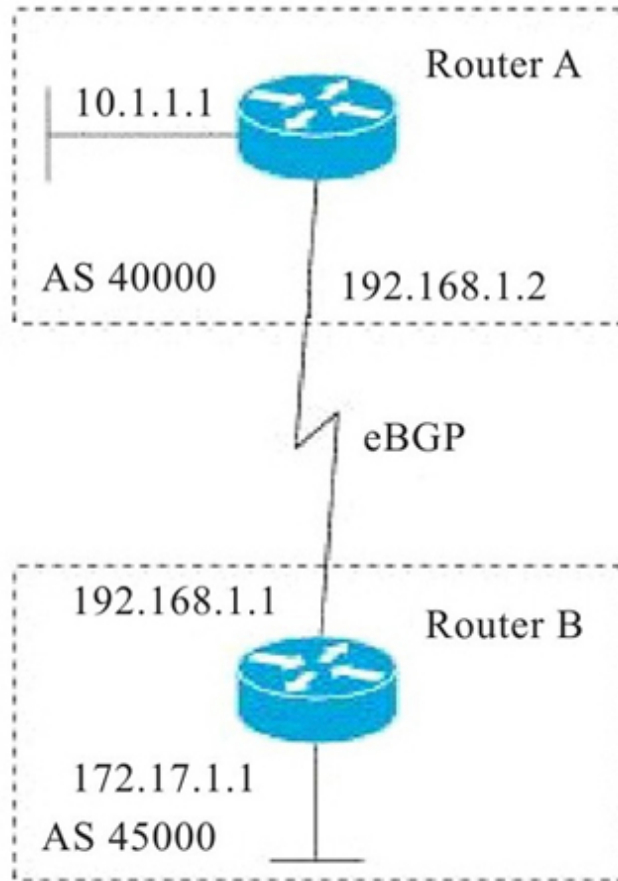


图 9-45 带有两个AS的简单BGP拓扑示例

经验之谈 这里要说明一点，在BGP通信中，对等路由器的AS号是不同的，在Internet上的公用AS号也必须是整个Internet上唯一的，因为它是由IANA（Internet Assigned Number Authority，互联网数字分配机构）全球统一分配的，就像公网IP地址。私有AS号就像局域网中私有IP地址一样，各种专用网络可以独立使用，不同专用网络可以重复使用，但不能在公用的Internet上分配使用。

1.BGP AS号格式

在2009年1月之前，RFC 4271中定义可以使用的BGP AS号是一个2字节数，取值在1~65 535范围之内。为了满足日益提高的AS号需求，IANA从2009年1月开始在RFC 5396中定义了4个字节的AS号，取值范围从65 536到4 294 967 295。AS有以下两种表示格式：

□Asplain（无格式AS）：一个十进制记数法所表示的十进制值，可以是2字节的，也可以是4字节的。如65 526是一个2字节的AS号，而234 567是一个4字节的AS号。

□Asdot（点分AS）：这种格式AS号是一个点分记数法所表示的十进制值。如果是2字节的AS号，则直接用它的十进制表示，如果是4字节的AS号，则采用点分计数法。例如65 526是一个2字节的AS号，仍采用65 526表示，而234 567是一个4字节的AS号，则表示为1.169031。计算方法是先把这个十进制转换成二进制（结果为11010001111100111），然后每十六位（2字节）分成一段（分别得到1和1010001111100111），两段之间以小圆点分隔，再对这两段分别换算成十进制即是了。

在一些品牌设备中（如Cisco IOS 12.0（32）S12、12.4（24）T和以后版本的路由器中）4字节的AS号仅以asdot格式显示，如1.10或者45 000.64000。在使用正则表达式来匹配asdot格式AS时，则在表达式中包括一个用来表示句点（.）的指定字符，通常是以反斜杠（\）表示。表

9-6显示了在仅asdot格式可用时的2字节和4字节AS号的配置和显示格式。

表 9-6 仅 asdot 格式可用时 2 字节和 4 字节 AS 号的配置和显示格式

格式	配置格式	输出和正则表达式匹配格式
asdot	2 字节 : 1 to 65535	2 字节 : 1 to 65535
	4 字节 : 1.0 to 65535.65535	4 字节 :1.0 to 65535.65535

2.保留和私有的AS号

在RFC 4893中，23 456是保留的AS号，它不能在用来配置AS号。后来，又在新的RFC 5389中规定了新的保留AS号，它们是在644 956~64 511之间的2字节AS号和在65 536~65 551之间的4字节AS号，它们也不能用来配置。

在64 512~65 534之间的2字节AS号是私有的，65 535是保留用于特定用途的。私有AS号可以用于内部路由域（相当于局域中使用的私有IP地址一样），但不能传输到达Internet的通信。BGP不应当配置通告私有AS号到外部网络。

9.4.3 BGP地址簇模型

BGP AFI（Address Family Identifier，地址簇标识符）模型是随多协议BGP而引入的，设计用来模块化，且具有可伸缩性，支持多AFI和SAFI（Subsequent Address Family Identifier，子序列地址族标识符）配置。网络的复杂性日益提高，许多公司正在使用BGP连接许多个AS，如图9-46所示。图中每一个独立的AS可能运行几种路由，如MPLS（Multiprotocol Label Switching，多协议标签交换）和IPv6路由，还可能需要通过BGP同时支持单播和组播路由。

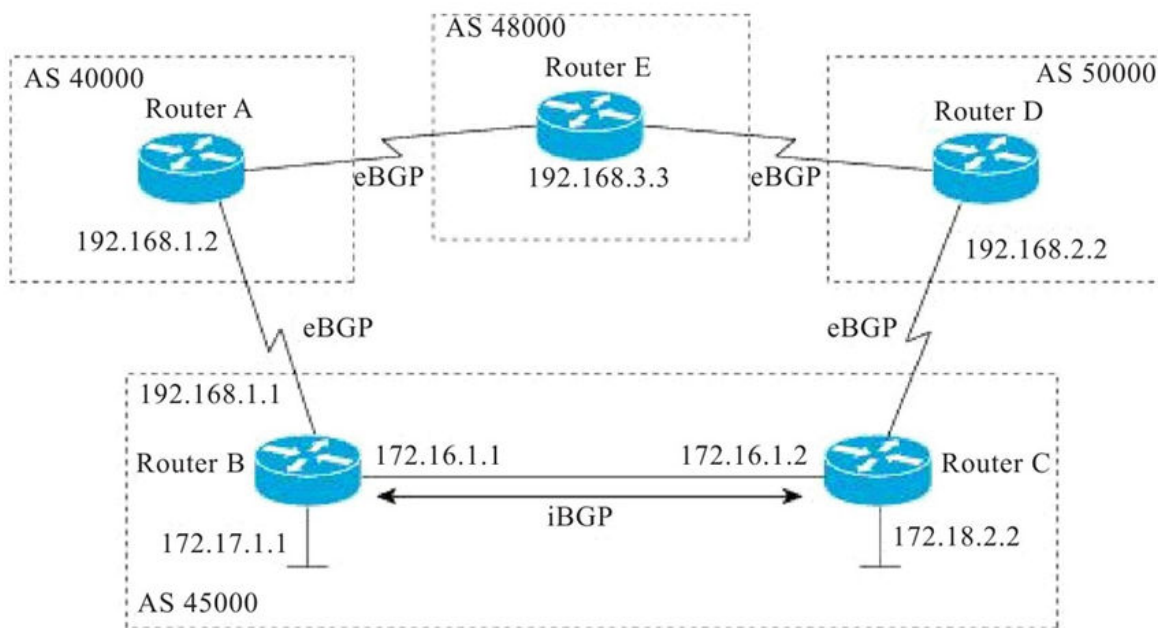


图 9-46 带有多地址簇的BGP网络拓扑示例

多协议BGP可以为多个网络层协议和IP组播路由承载路由信息。这个路由信息作为BGP附加属性（多协议扩展属性）加载在AFI模型之中。每个地址簇维护一个独立的BGP数据库，允许基于每个地址簇配置BGP策略。SAFI配置父AFI的子集，用于精确BGP策略配置。

AFI模型是因为NLRI格式的弹性局限性而创建的，以NLRI格式配置的路由器可以有IPv4单播，但限制了组播特性。在NLRI格式中配置的网络有以下几方面的局限性：

- 没有支持AFI和SAFI配置信息。许多新的BGP功能（如MPLS）仅可在AFI和SAFI配置模式下配置，不能在NLRI配置模式下配置。

- 不支持IPv6。在NLRI格式配置中的路由器不能与IPv6邻居建立连接。

- 限制组播域间路由和非一致的组播和单播拓扑支持。在NLRI格式中，并不是所有选项都是可用的，并且不支持VPNv4。NLRI格式配置可以比AFI模型配置更复杂。如果在基础架构中的路由器不支持组播功能，或者策略与配置用于组播通信的策略存在差异，则组播路由将不被支持。

在多协议BGP中的AFI模型支持多个AFI和SAFI，以及所有基于NLRI的命令和策略配置，并且向后与仅支持NLRI格式的路器由兼容。配置使用AFI模型的路由器具有以下功能特征：

□支持AFI、SAFI信息和配置。配置使用AFI模型的路由器可以承载多个网络层协议地址簇（如IPv4和IPv6）路由信息。

□所有地址簇中的AFI配置是相似的，而且在语法格式上比NLRI格式语法更容易。

□支持所有BGP路由功能和命令。

□支持不同策略的一致或非一致单播和组播拓扑。

□支持CLNS（无连接网络服务）。

□支持在仅支持NLRI格式（向后兼容基于AFI的网络）的路由器间相互操作，这包括同时支持IPv4单播peer和组播NLRI peer。

□支持VPN和VRF（VPN路由和转发）实例。VRF的单播IPv4地址可以在指定的IPv4 VRF地址簇下配置，这种配置更新是集成到BGP VPNv4数据库中的。

在一个指定的地址簇配置模式下，可以使用“？”查显示当前可以使用的命令。相同BGP功能既可以在地址簇配置模式下配置，又可以在路由器配置模式下配置，但是在路由器配置模式下的BGP命令仅适用于IPv4单播地址前缀。要为其他地址簇前缀（如IPv4组播或者IPv6单播地址前缀）使用BGP命令和功能，必须进入对应地址簇配置模式下进行配置。

BGP地址簇配置模式包括四个地址簇：IPv4、IPv6、CLNS和VPNv4。有些设备还支持L2VPN地址簇，在L2VPN地址簇内支持VPLS SAFI。在IPv4和IPv6地址簇SAFI内存在MDT（组播分布树）、隧道和VRF。表9-7所示为不同SAFI（子序列地址簇标识符）所对应的字段值及功能说明。为了确保运行所有AFI和SAFI配置的网络间兼容，建议在设备上使用多协议BGP地址簇模型配置BGP。

表 9-7 SAFI 类型及功能说明

SAFI 字段值	描 述	参考文档
1	用于单播转发的 NLRI	RFC 2858

(续)

SAFI 字段值	描 述	参考文档
2	用于组播转发的 NLRI	RFC 2858
3	可同时用于单播和组播转发的 NLRI	RFC 2858
4	带有 MPLS 标签的 NLRI	RFC 3107
64	隧道 SAFI	draft-nalawade-kapoor-tunnel-safi -01.txt
65	VPLS (Virtual Private LAN Service, 虚拟专用局域网服务)	—
66	BGP MDT SAFI	draft-nalawade-idr-mdt-safi-00.txt
128	带有 MPLS 标签的 VPN 地址	RFC-ietf-l3vpn-rfc2547bis-03.txt

9.4.4 BGP speaker和peer的关系

在配置BGP路由中，我们见得最多的可能就是peer这个词了，它代表了什么？它与我们以前在其他路由协议中经常见到的“邻居”，以及在BGP路由配置中经常见到的speaker是什么关系？这是学习BGP路由协议的一个关键所在。

1.BGP-speaking、BGP speaker和peer的关系

在BGP网络中，所有参与BGP进程的路由器都称为BGP-speaking路由器（BGP-speaking可以看成是BGP会话的意思）。每个BGP-speaking路由器不能自动发现其他BGP-speaking设备，通常需要网络管理员手动配置BGP-speaking间的关联。对于活动的BGP-speaking设备，称为peer设备，它与其他BGP-speaking设备之间有一个活动的TCP连接。

BGP设备之间的这种peer关系通常是指邻居关系（和RIP、OSPF等网络中的邻居设备关系一样），也就是指这两个设备之间没有通过其他路由器进行连接。但在BGP中，直接连接的BGP路由器的关系不是邻居关系，而直接用peer（对等）关系描述。

BGP speaker是指本地BGP路由器，而peer（对等，或者对端）是指任何其他BGP-speaking网络设备。当然这种关系是相对的，就看当前是在哪台路由器上配置了。也可以这么理解，BGP-Speaking路由器是

包含了已与其他路由器之间建立了BGP连接，或者没有与其他路由器建立BGP连接的所有BGP路由器的总称，BGP speaker是指自己当前所在的BGP路由器，而peer是指所有与其他路由器之间建立了TCP连接的BGP路由器的总称。但是一个BGP路由器可以有多个邻居路由器，也就相当于可能有多多个peer路由器。

现假设有一个图9-47所示的网络拓扑，包含Router A、Router B、Router C、Router D和Router E五个都运行了BGP路由协议的路由器。其中Router A与Router B、Router E之间建立了TCP连接，而与Router C之间还没有建立TCP连接，目前正在配置的是Router A。这时，所有这五个路由器都是BGP-speaking设备，Router A是BGP speaker，Router B和Router E对于Router A来说是peer设备，但Router C和Router D只是BGP-speaking设备，而不是peer设备，因为Router C没有与Router A建立TCP连接，而Router D不是直接与Router A连接的，也就是不是邻居关系。而且，Router B与Router E也不是peer关系，因为它们之间没有直接连接，尽管它们都与Router A有直接连接。

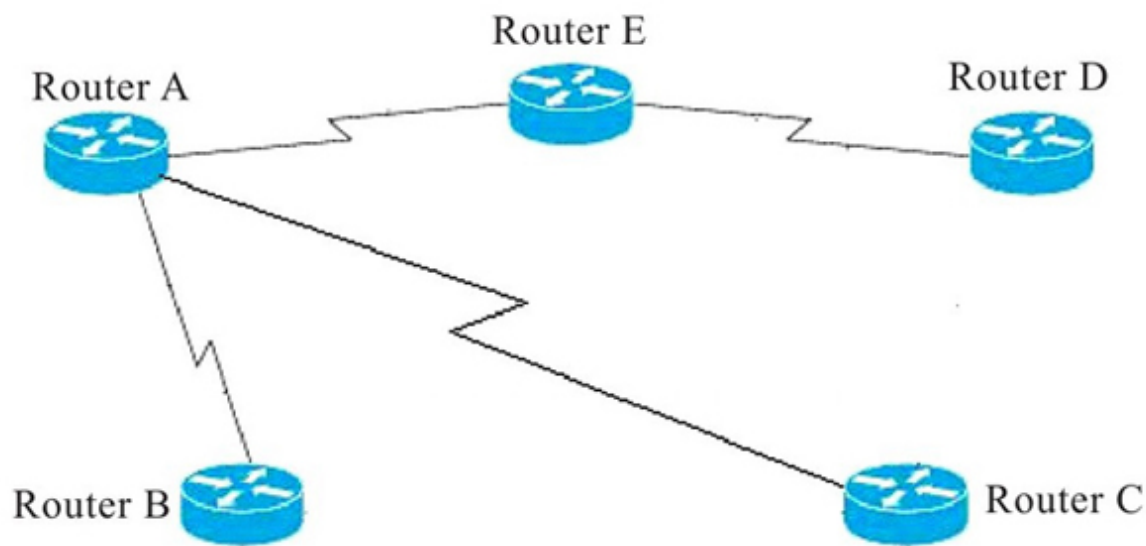


图 9-47 BGP speaker与peer的关系示意图

当在peer设备间（也就是相互直接连接的BGP路由器之间）建立了TCP连接，每个BGP peer就会立即与对端交换所有的路由表，也就是完整的BGP路由表。在第一次交换后，后面的路由信息交换在网络拓扑发生更改，或者完成或者修改了一个路由策略时仅发送增量更新。在长时间没有路由更新发送的时间段内，两BGP路由器对端也会交换一个称为keepalive（存活）的特殊消息，以告诉对方自己仍然正常工作，并没有发生拓扑更改。

2.内部peer与外部peer

BGP AS与我们在学习其他路由协议中的AS是一样的概念，只是针对的路由协议不同而已。一个AS就是一个由单一技术管理（也就是它可以单独被管理）实体控制的网络，对应一个BGP路由进程。当BGP

peer路由器位于不同AS中时，它们之间互称对方为外部peer，当它们位于同一个AS中时，则称为内部peer。通常，外部peer是邻接并共享一个子网的，内部peer可以在同一个AS中的任何地方。

在图9-48所示的示例中，Router B与Router C就是内部peer关系，它们同在AS 45000中，并且通过共享172.16.1.0/24这个子网相邻连接。而Router A与Router B、Router E之间，Router E与Router D之间，Router D与Router C之间都是外部peer关系，因为它们各自不在同一AS中。内部peer之间的连接是通过iBGP协议进行的，而外部peer之间的连接则需要通过eBGP协议进行。在具体的路由器BGP协议配置时一定要注意正确选择使用。

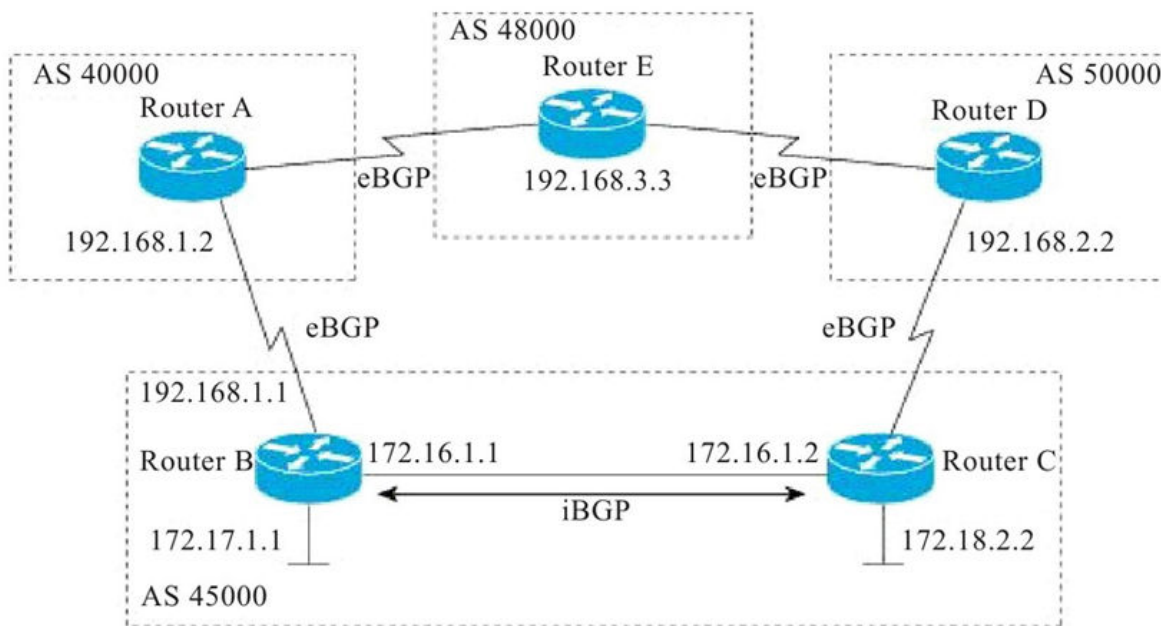


图 9-48 内/外部peer关系示意图

9.4.5 BGP peer会话建立

当一个BGP路由进程与对端建立对等会话时，将按顺序完成以下状态更改：

1) 空闲 (Idle)：这是在路由进程启动或者路由器复位时，进入路由进程的初始状态。在这个状态中，路由器处于等待状态，等待一个启动事件 (start event，如与一个远程peer路由器进行对等会话配置) 的发生。当这个路由器接收到一个来自远程peer的TCP连接请求时，路由器会发起其他启动事件，但在发起一个TCP连接到远程peer之前会等待一个计时器到期。如果这个路由器复位，则peer路由器也复位，这个BGP路由进程返回到空闲状态。

2) 连接 (Connect)：BGP路由进程检测到一个peer正在试图与本地BGP speaker建立一个TCP会话。

3) 活动 (Active)：在这个状态下，本地BGP speaker的BGP路由进程试图使用重试连接计时器与一个peer路由器建立一个TCP会话。但当BGP路由进程处于活动状态时，这种来自远程peer的启动事件是被忽略的，也就是本地BGP speaker不会响应peer的TCP重试连接请求。如果路由进程被重新配置或者发生错误，BGP路由进程将释放系统资源，返回到最初的空闲 (idle) 状态。

4) 打开发送 (OpenSent) : 在TCP连接建立好后, BGP路由进程发送一个Open消息到远程peer, 并转换进入到打开发送 (OpenSent) 状态。在这个状态下, BGP进程可以接收其他peer的Open消息。如果连接失败, BGP路由进程转换到上面的活动 (Active) 状态。

5) 打开接收 (OpenReceive) : 在自己发送完Open消息后, 就同时进入打开接收 (OpenReceive) 状态。此时, BGP路由进程可从远程peer接收Open消息, 等待一个来自远程peer的初始keepalive消息。当接收了一个keepalive消息, BGP路由进程转换进入到下面将要介绍的建立 (Established) 状态; 如果接收了一个通知消息, BGP路由进程则转换进入到最初的空闲 (idle) 状态; 如果发生错误或者发生了影响 peering 会话的配置更改, 则BGP路由进程以FSM (Finite State Machine, 有限状态机制) 错误代码发送一个通知消息, 然后进入到最初的空闲 (idle) 状态。

6) 建立 (Established) : 当从远程peer接收到了一个初始 keepalive消息, 即表明正在与远程邻居建立会话, BGP路由进程也开始与远程peer交换更新消息。当接收了一个更新或者keepalive消息时, 连接保持计时器 (hold timer) 也开始计时了 (具体连接保持的时间有多长是根据保持计时器的配置而定的)。如果BGP进程收到一个错误通知, 则路由器将转换进入到最初的空闲 (idle) 状态。

9.4.6 BGP的路由属性

BGP路由属性是一组参数，对特定的路由进行了进一步的描述，使得**BGP**能够对路由进行过滤和选择。所有的**BGP**路由属性都可以分为以下四类：

□公认必须遵循（**Well-known mandatory**）：所有**BGP**路由器都必须能够识别这种属性，且必须存在于**Update**（更新）消息中。如果缺少这种属性，路由信息就会出错。

□公认可选（**Well-known discretionary**）：所有**BGP**路由器都可以识别，但不要求必须存在于**Update**消息中，可以根据具体情况来选择。

□可选过渡（**Optional transitive**）：在**AS**之间具有可传递性的属性。**BGP**路由器可以不支持此属性，但它仍然会接收带有此属性的路由，并通告给其他对等体。

□可选非过渡（**Optional non-transitive**）：如果**BGP**路由器不支持此属性，该属性被忽略，且不会通告给其他对等体。

表9-8所示为**BGP**路由的几种基本属性与它们对应的属性类别。

表 9-8 路由属性及对应的类别

属 性	类 别
ORIGIN	公认必须遵循
AS_PATH	公认必须遵循
NEXT_HOP	公认必须遵循
LOCAL_PREF	公认可选
ATOMIC_AGGREGATE	公认可选
AGGREGATOR	可选过渡
COMMUNITY	可选过渡
MULTI_EXIT_DISC (MED)	可选非过渡
ORIGINATOR_ID	可选非过渡
CLUSTER_LIST	可选非过渡

下面介绍表9-8中几种最主要的BGP路由属性。理解这些路由属性对于理解BGP路由工作原理非常重要。

1.源（ORIGIN）属性

ORIGIN属性定义路由信息的来源，也就是标记一条路由是怎么成为BGP路由的。它有以下三种可能：

□IGP：优先级最高，说明路由产生于本AS内。

□EGP：优先级次之，说明路由通过EGP学到。

□incomplete：优先级最低，但并不是说明路由不可达，而是表示路由的来源无法确定。例如，引入的其他路由协议的路由信息。

2.AS路径（AS_PATH）属性

AS_PATH属性按一定次序记录了某条路由从本地到目的地址所要经过的所有**AS**号。当**BGP**将一条路由通告到其他**AS**时，便会把本地**AS**号添加到**AS_PATH**列表的最前面。收到此路由的**BGP**路由器根据**AS_PATH**属性就可以知道去目的地址所要经过的**AS**。离本地**AS**最近的相邻**AS**号排在前面，其他**AS**号按顺序依次排列。图9-49所示是从**AS 10**中的路由器分别向**AS 20**和**AS 40**中的路由器发送一条路由通告时的**AS_PATH**属性（括号中的值，如30，20，10）变化的示例。

通常情况下，**BGP**不会接受**AS_PATH**中已包含本地**AS**号的路由，从而避免了形成路由环路的可能。同时，**AS_PATH**属性也可用于路由的选择和过滤。在其他因素相同的情况下，**BGP**会优先选择路径较短的路由。比如在图9-49中，**AS 50**中的**BGP**路由器会选择经过**AS 40**的路径作为到目的地址8.0.0.0的最优路由。

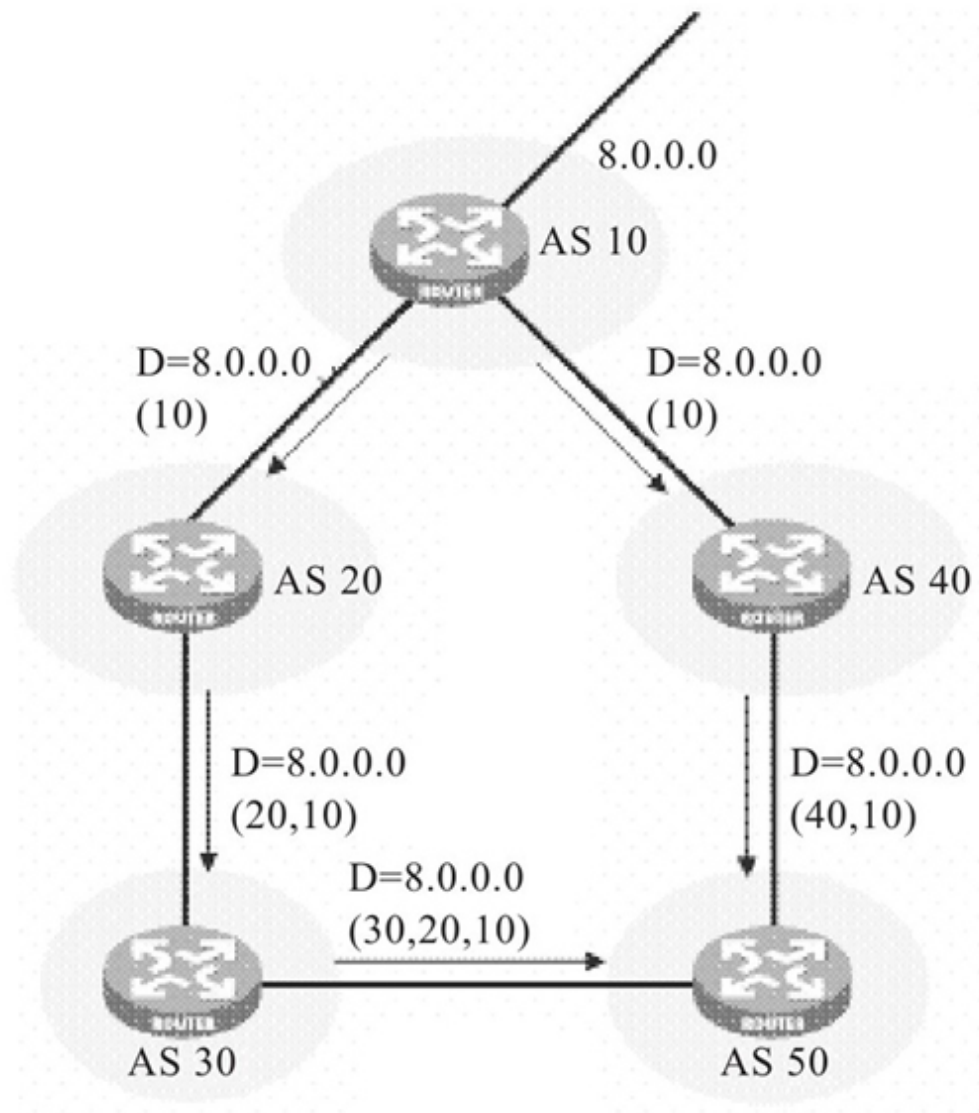


图 9-49 BGP路由AS_PATH属性示例

在某些应用中，可以使用路由策略来人为地增加AS路径的长度，以便更为灵活地控制BGP路径的选择。通过AS路径过滤列表，还可以针对AS_PATH属性中所包含的AS号来对路由进行过滤。

3. 下一跳（NEXT_HOP）属性

BGP的下一跳属性不一定是邻居路由器的IP地址，它的取值为以下三种情况：

1) 当BGP发言者把自己产生的路由发给所有邻居时，将把该路由信息的下一跳属性设置为自己与对端连接的接口地址。

2) 当BGP发言者把接收到的路由发送给eBGP对等体时，将把该路由信息的下一跳属性设置为本端与对端路由器连接的接口地址。

在如图9-50中，AS 200中的路由器向AS 300中的路由器发送路由时就把下一跳设为本端与对端路由器连接的接口地址1.1.2.1。图中AS 100中的路由器向AS 200中的路由器发送路由通告时的下一跳设置也是如此。

3) 当BGP发言者把从eBGP邻居得到的路由发给iBGP邻居时，并不改变该路由信息的下一跳属性。如果配置了负载分担，路由被发给iBGP邻居时则会修改下一跳属性。

在图9-50中，AS 300中的路由器在收到AS 200路由器发来的路由通告时，再在内部路由器之间转发时，下一跳地址仍为原来的1.1.2.1。

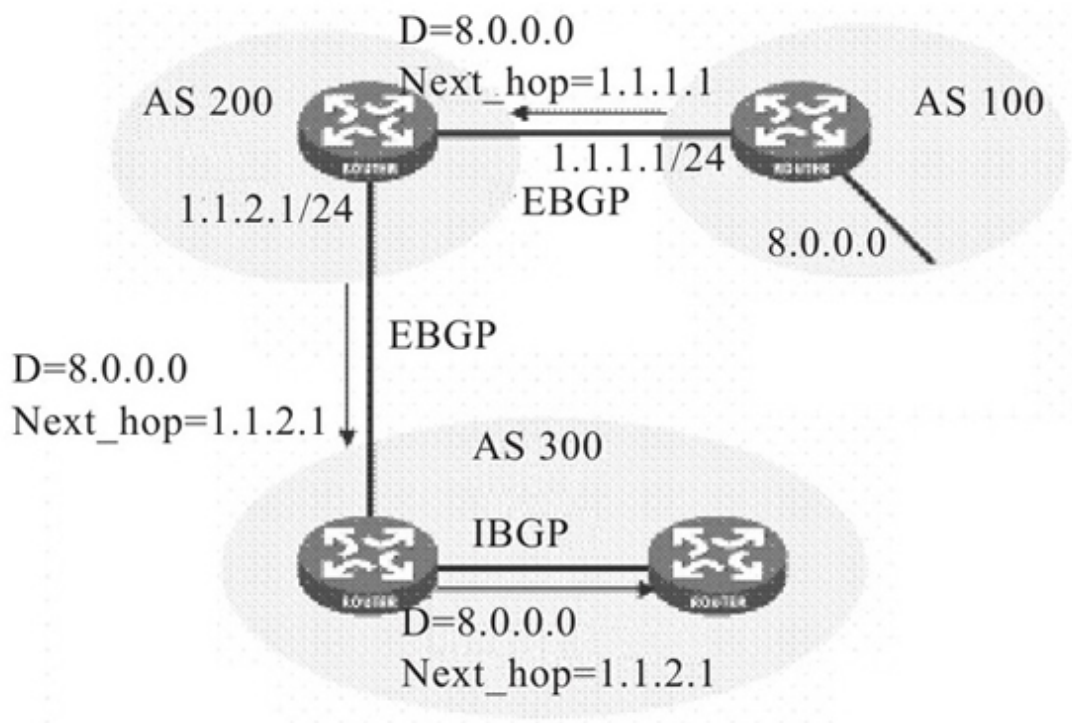


图 9-50 BGP路由下一跳属性示例

4.MED (MULTI_EXIT_DISC)

MED（多出口度量）属性相当于我们前面介绍RIP、OSPF协议时所说的路由度量值，它用于判断流量进入AS时的最佳路由，也就是控制流量从哪条路径进入AS。当一个运行BGP的路由器通过不同的eBGP对等体得到目的地址相同但下一跳不同的多条路由时，在其他条件相同的情况下，将优先选择MED值较小者作为最佳路由。如图9-51所示，从AS 10到AS 20的流量将选择Router B作为入口。

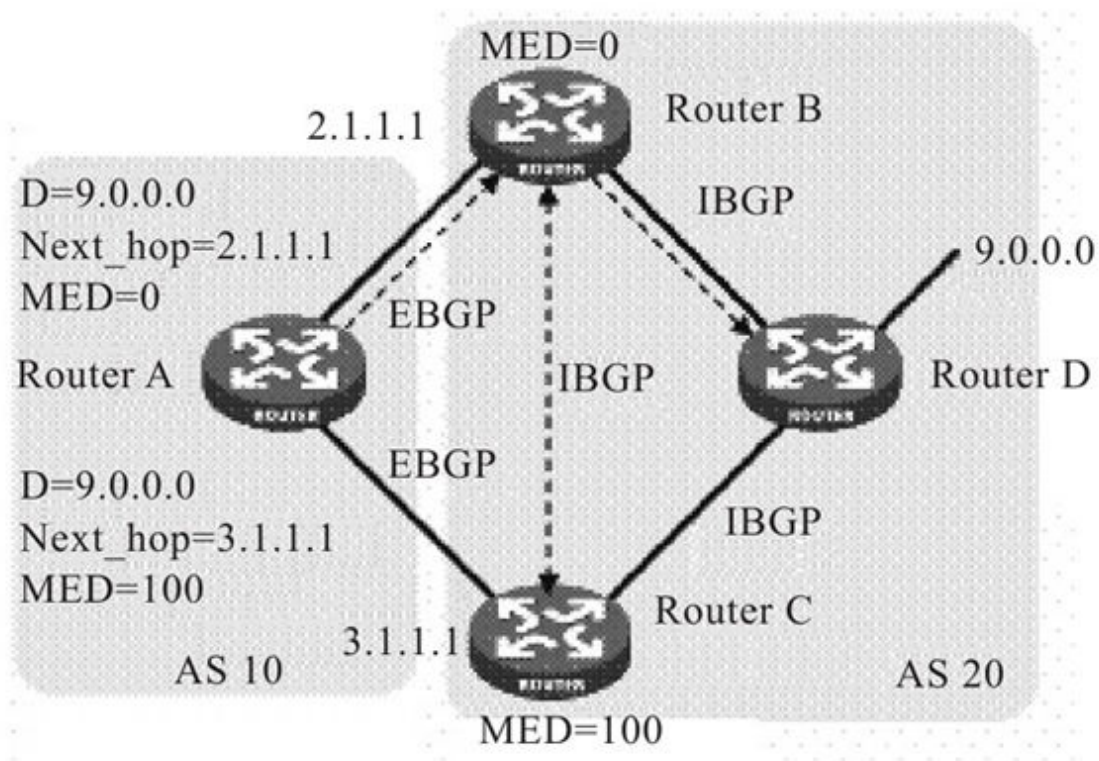


图 9-51 BGP路由器MED属性示例

MED属性仅在相邻两个AS之间交换，收到此属性的AS一方不会再将其通告给任何其他第三方AS。通常情况下，BGP只比较来自同一个AS路由的MED属性值。

5.本地优先（LOCAL_PREF）属性

LOCAL_PREF属性用于判断流量离开AS时的最佳路由，与上面介绍的MED属性所控制的正好相反。当BGP的路由器通过不同的IBGP对等体得到目的地址相同但下一跳不同的多条路由时，将优先选择

LOCAL_PREF属性值较高的路由。如图9-52所示，从AS 20到AS 10的流量将选择Router C作为出口。

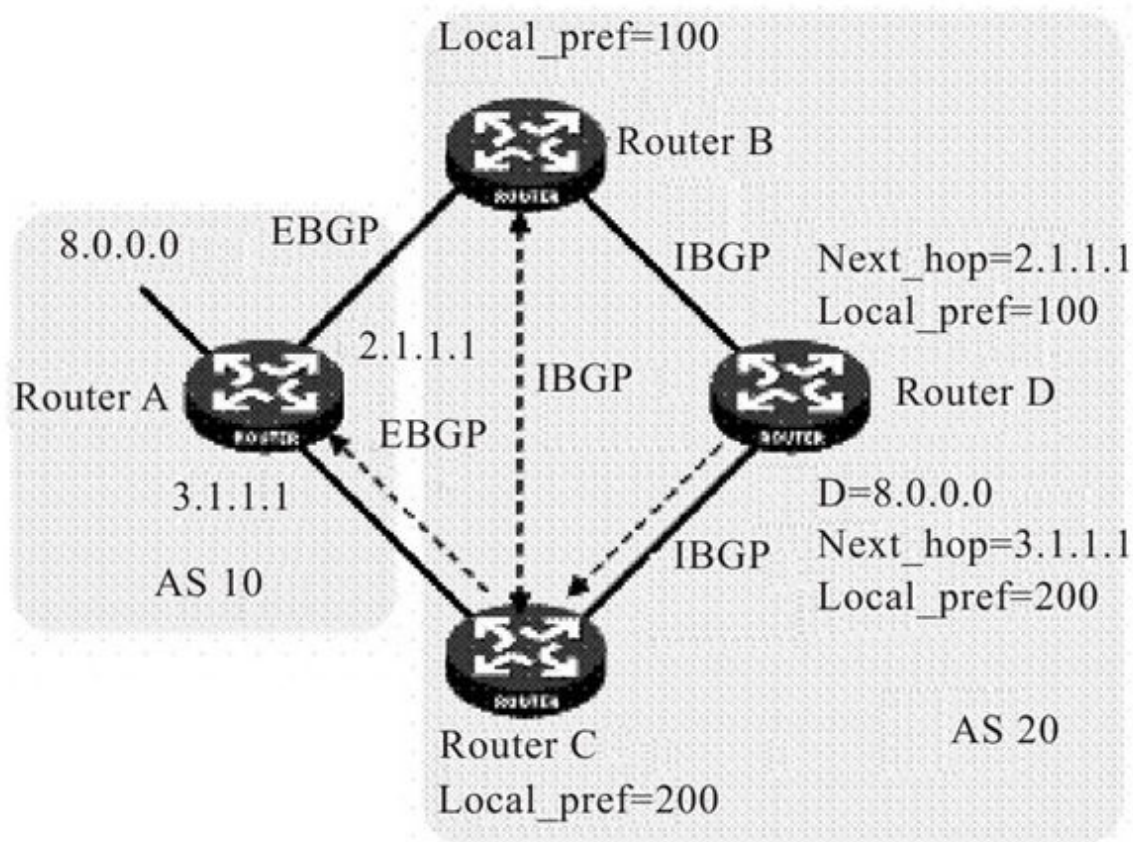


图 9-52 LOCAL_PREF属性

LOCAL_PREF属性仅在iBGP对等体之间交换，不通告给其他AS。它表明BGP路由器的优先级。

6.团体（COMMUNITY）属性

团体属性用来简化路由策略的应用和降低维护管理的难度。它是一组有相同特征的目的地址的集合，没有物理上的边界，与其所在的

AS无关。公认的团体属性有：

☐INTERNET：默认情况下，所有的路由都属于INTERNET团体。具有此属性的路由可以被通告给所有的BGP对等体。

☐NO_EXPORT：具有此属性的路由在收到路由更新后，不能被发布到本地AS之外。如果使用了联盟，则不能被发布到团体之外，但可以发布给团体中的其他子AS。

☐NO_ADVERTISE：具有此属性的路由被接收后，不能被通告给任何其他BGP对等体。

☐NO_EXPORT_SUBCONFED：具有此属性的路由被接收后，不能被发布到本地AS之外，也不能发布到团体中的其他子AS。

9.4.7 BGP的消息类型及报文格式

BGP协议有五种消息类型：Open（建立）、Update（更新）、Notification（通知）、Keepalive（保持活跃）和Route-refresh（路由刷新）。这些消息有相同的报文头，其格式如图9-53所示。

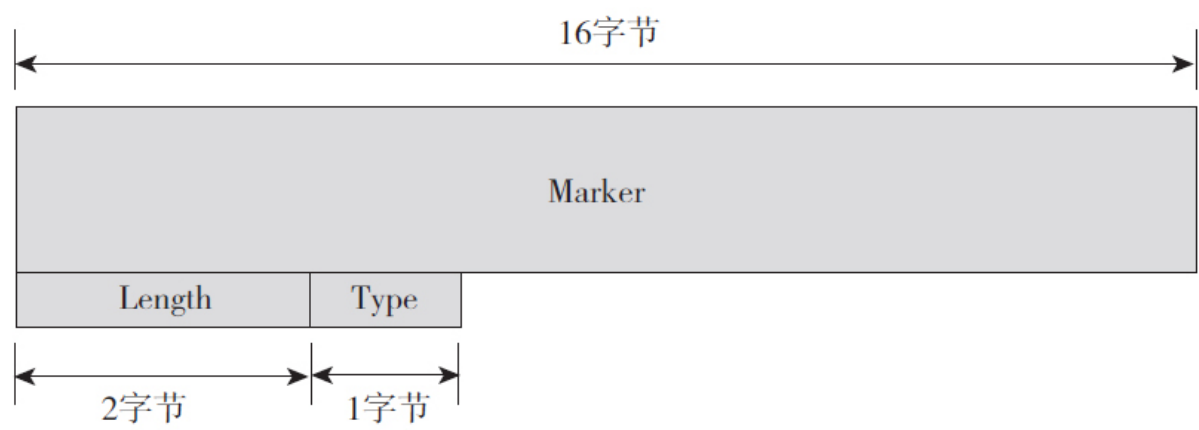


图 9-53 BGP消息的报头格式

图9-53所示各字段解释如下：

□Marker: 占16字节，用于标明BGP报文边界，固定值为所有位均为1，相当于一个报文的头部标识符。

□Length: 占2字节，标识BGP消息总长度（包括报头在内），以字节为单位。

□Type: 占1字节，标识BGP消息的类型。其取值从1到5，分别表示Open、Update、Notification、Keepalive和Route-refresh消息。其中，前四种消息是在RFC 1771中定义的，而Type为5的消息是在RFC 2918中定义的。

下面再具体介绍这五种消息报文格式。

1.Open消息报文格式

Open（建立）消息是TCP连接建立后发送的第一个消息，用于建立BGP对等体之间的连接关系。其消息的报文格式如图9-54所示。

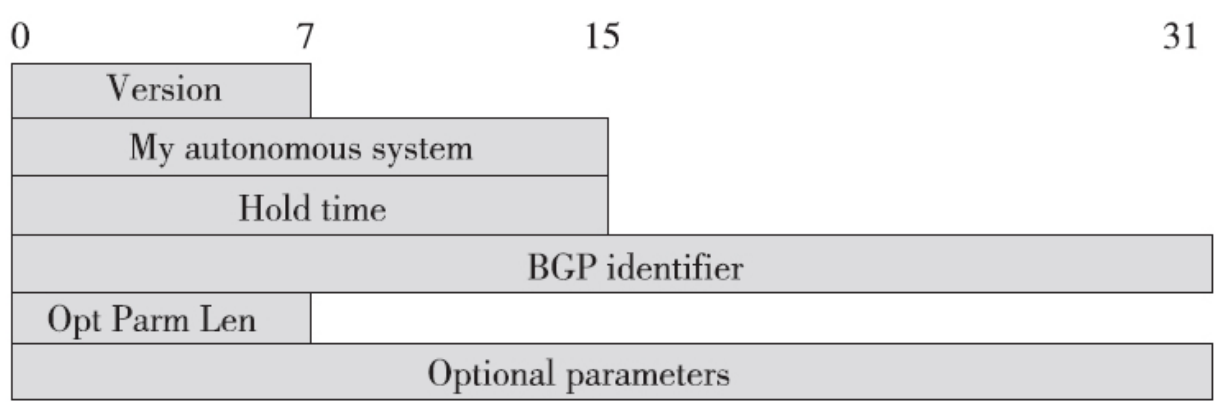


图 9-54 BGP Open消息报文格式

图9-54所示各字段解释如下：

□Version: 版本，占1字节，标识BGP版本号。对于BGPr4来说，其值为4。

□My autonomous system: 本地AS字段，占2字节，标识本地AS号。通过比较两端的AS号可以确定是eBGP连接还是iBGP连接。

□Hold time: 保持时间，占2字节，以秒为单位。在建立对等体关系时两端要协商Hold time，并保持一致。如果在这个时间内未收到对端发来的Keepalive消息或Update消息，则认为BGP连接中断。

□BGP identifier: BGP标识符，占4字节，以IP地址的形式标识BGP路由器ID，用来识别BGP路由器。

□Opt Parm Len (Optional Parameters Length): 可选参数的长度，占1字节，标识可选参数的总长度，如果为0则没有可选参数。

□Optional parameters: 可选参数，长度可变，用于多协议扩展 (Multiprotocol Extensions) 等功能。

2.Update消息报文格式

Update (更新) 消息用于在对等体之间交换路由信息。它既可以发布可达路由信息，也可以撤销不可达路由信息。其消息报文格式如图9-55所示。

Unfeasible routes length	2 Octets
Withdrawn routes	N Octets
Total path attribute length	2 Octets
Path attributes	N Octets
NLRI	N Octets

图 9-55 BGP Update消息报文格式

一条Update报文可以通告一类具有相同路径属性的可达路由，这些路由放在NLRI（Network Layer Reachable Information，网络层可达信息）字段中，Path Attributes字段携带了这些路由的属性，BGP根据这些属性进行路由的选择；同时Update报文还可以携带多条不可达路由，被撤销的路由放在Withdrawn Routes字段中。

图9-55所示各字段解释如下：

□Unfeasible routes length: 不可达路由字段的长度，占2字节，以字节为单位。如果为0则说明没有Withdrawn routes字段。

□Withdrawn routes: 不可达路由的列表，可变长度。

□Total path attribute length: 路径属性字段的长度，占2字节，以字节为单位。如果为0则说明没有Path attributes字段。

□Path attributes: 与NLRI相关的所有路径属性列表，每个路径属性由一个TLV（Type-Length-Value）三元组构成，可变长度。BGP正是根据这些属性值来避免环路、进行选路、协议扩展。

□NLRI（Network Layer Reachability Information）：可达路由的前缀和前缀长度二元组，可变长度。

3.Notification消息及报文格式

当BGP检测到错误状态时，就向对等体发出Notification（通知）消息，之后BGP连接会立即中断。其消息报文格式如图9-56所示。

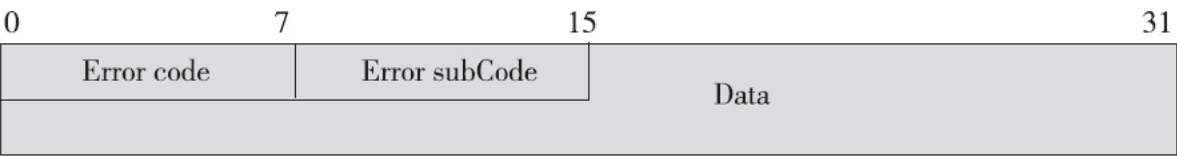


图 9-56 BGP Notification消息报文格式

图9-55所示各字段解释如下：

□Error code: 差错码，占1字节，指定错误类型。

□Error subcode: 差错子码，占1字节，描述错误类型的详细信息。

□Data: 错误消息内容，可变长度，用于辅助发现错误的原因，它的内容依赖于具体的差错码和差错子码，记录的是出错部分的数据。

4.Keepalive消息报文格式

BGP会周期性地向对等体发出Keepalive（保持活跃）消息，用来保持连接的有效性。其消息格式中只包含图9-53所示的BGP消息报头，没有附加其他任何字段。

5.Route-refresh消息报文格式

Route-refresh（路由刷新）消息用来要求对等体重新发送指定地址簇的路由信息。其消息格式如图9-57所示。

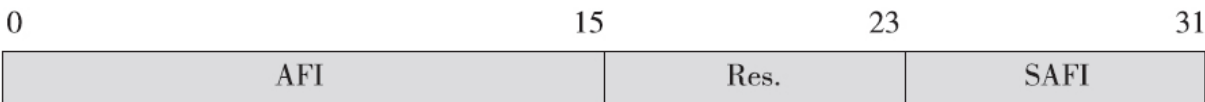


图 9-57 BGP Route-refresh消息报文格式

图9-57所示各字段解释如下：

□AFI: Address Family Identifier，地址簇标识，占2字节，用于标识所采用的地址簇类型。

□Res.: 保留，占1字节，必须置0。

□SAFI: Subsequent Address Family Identifier，子地址簇标识，占1字节，用于标识子地址簇类型。

在多协议BGP中支持多种三层地址簇，如IPv4、IPv6、CLNS、VPNv4和L2VPN地址簇。AFI和SAFI两个字段是多协议BGP中用来区分不同网络层协议的地址簇编号。常见的地址簇和子地址簇标识如表9-9所示。

表 9-9 常用地址簇类型的 AFI 和 SAFI

BGP 地址簇类型	AFI	SAFI
IPv4 单播	1	1
IPv4 组播	1	2
IPv4 Label	1	4
IPv4 VPNv4	1	128
Ipv6 单播	2	1
IPv4 MDT（组播分布树）	1	66
IPv6 组播	2	2
L2VPN	196	128
VPLS（rfe4761）	25	65

第10章 传输层

“传输层”与“数据链路层”的作用都是建立数据传输通道，两者在功能上存在许多相似之处。传输层的功能主要体现在广域网网络应用中，可以把“数据链路层”当做局域网通信的数据传输通道，而“传输层”则是广域网中的数据传输通道。本章主要针对广域网中的传输层来进行各方面功能原理的介绍。但是，在局域网中的网络应用同样需要用到传输层功能，也是由用户计算机操作系统中TCP或UDP来完成的，不同之处在于，在局域网中，传输连接是永久连接，因为局域网中的数据链路是永久连接的，又不需要与其他网络进行连接。

“传输层”是整个广域网网络体系结构模型的核心所在，因为它负责端到端的通信，是面向网络通信的低三层和面向信息处理的高三层之间的中间一层，起到桥梁作用。“传输层”同时是两台计算机系统经过网络进行数据通信时第一个端到端的层次。本章主要介绍OSI/RM中的传输层协议和TCP/IP体系结构中TCP的各种功能（主要包括传输连接建立/释放、数据传输、差错控制、流量控制、拥塞控制等）的实现原理。

10.1 传输层概述

在整个网络体系结构中，通常是将OSI/RM七层模型中的下面三层称为面向通信子网的层，负责通信通道的建立，而将传输层及以上的各层称为面向资源子网的层，负责终端系统间的数据通信。还有一种划分方式，即将“传输层”及以下的三层统称为面向通信的层，总体来说，负责通信通道的建立和数据传输，而将“会话层”、“表示层”和“应用层”这些不包含任何数据传输功能的层统称为面向应用的层，如图10-1所示。

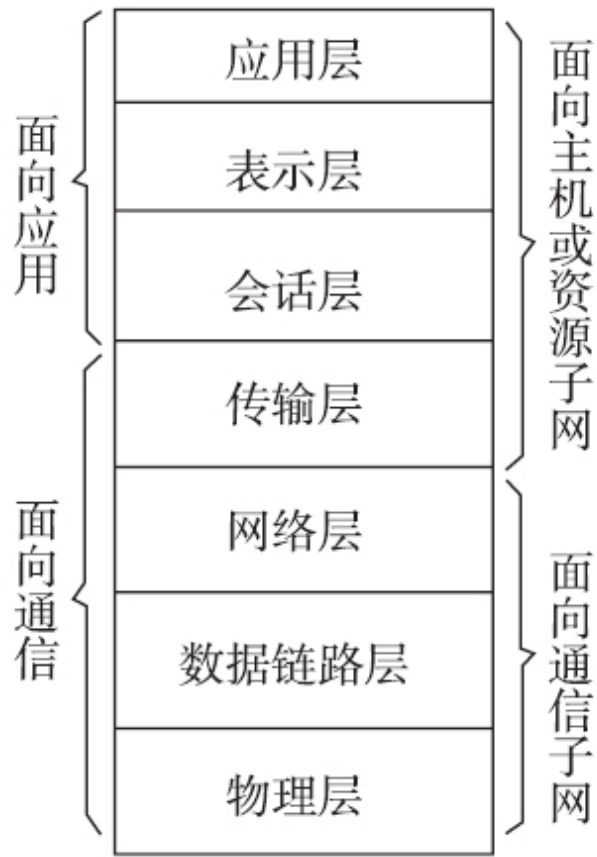


图 10-1 传输层在OSI/RM模型中的特殊位置

10.1.1 划分传输层的必要性

从通信和信息处理两方面来看，“传输层”既是面向通信部分的最高层，与下面的三层一起共同构建进行网络通信所需的线路和数据传输通道，同时又是面向用户的最低层，因为无论何种网络应用，最终都需要把各种数据报传送到对方。来自应用层的用户数据必须依靠传输层协议在不同网络中的主机间进行传输，因为仅靠网络层把数据传送到目的主机上还是不够的，还必须把它交给目的主机的应用进程。因此，无论“传输层”使用哪种划分方式，它的位置都非常特殊，都起到一个承上启下的桥梁作用。

下面再从数据通信原理方面具体分析各层的基本作用，从而体会划分“传输层”的必要性。物理层为数据通信提供实际的物理线路和通信信道，这是任何数据通信的基础；数据链路层为同一网络中（数据链路层的通信限于同一局域网中）的数据通信提供了虚拟的通信通道，可以根据不同链路类型对物理层的比特流进行帧封装和传输；网络层为不同网络间的数据通信提供了数据包的路由、转发功能，把数据包从一个网络中的主机传送到位于另一网络中的目的主机上，其中需要选择传送的最佳路径。这时，可能就有读者会问到，既然网络层已把源主机上发出的数据包传送给目的主机，那么为什么还需要设置一个传输层呢？这就需要我们理解主机间用户应用层通信的实体了。

位于两台网络主机间的真正数据通信主体不是这两台主机，而是两台主机中的各种网络应用进程。因为在同一时刻，两主机间可以进行多个应用通信。例如，某两个用户在进行视频通信时，还可以进行QQ聊天，甚至可以同时打网络电话，这就是多个应用进程，如图10-2所示（仅列出了一个方向）。而不同通信的标识就是进程，只有通信进程才可以把对应的数据包传输到对应的应用中。正因为如此，在运行具体的网络通信应用时，必须为每一个网络应用配备一个唯一的应用进程标识，否则所传输的报文就不知道要提交给哪个用户应用了。而这里的应用进程识别就要依靠本章所要介绍的“传输层”了，它就是通过“端口”将不同应用进程进行对应的。

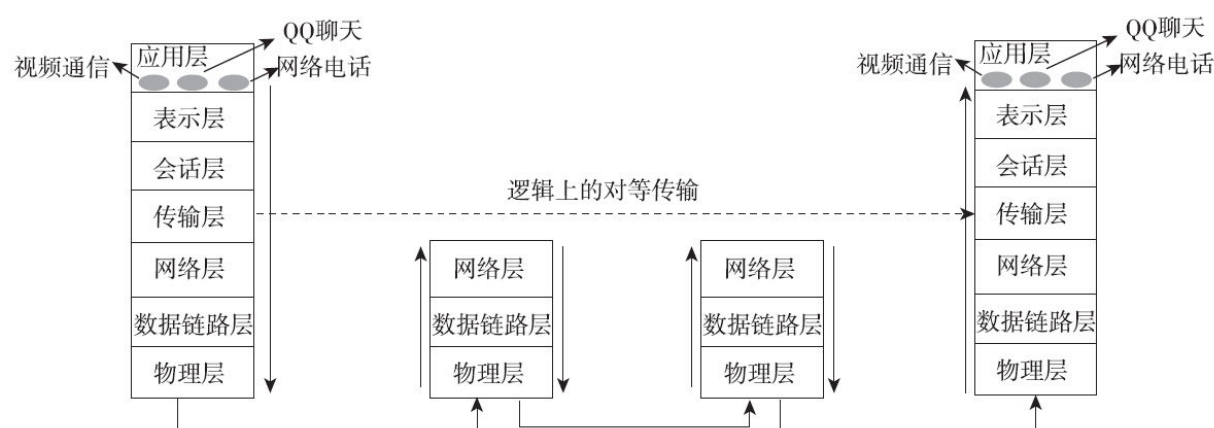


图 10-2 多应用进程的网络通信流程示意图

不同网络间主机的整体通信过程基本如下：在一个用户主机的应用层发出的应用请求报文到了传输层后，在数据的头部添加对应的传输层协议头部信息（具体内容将在本章后面介绍），将其封装成数据

段，然后传到网络层后封装在报文（或者报文分组）中的“数据”部分，再依次传到数据链路层，重新封装成数据帧，最后通过物理层以比特流的方式一位位地向对方网络传输。传送到对方网络中后，数据沿着与发送端相反的方向进行解封装，然后依次传输到对应的传输层，最后提交给应用层中的相应应用进程。如果中间经过多个路由器，则这些中间路由器只进行最低三层的报文封装和解封装，以及其他对应功能，不建立基于传输层的对等连接关系。

“传输层”是源端到目的端对数据传送进行控制的从低到高的最后一层，最终目标就是向它的用户（应用层中的应用进程）提供高效、可靠和性价比合理的服务。在传输层内部，完成这项任务的硬件或软件称为传输实体（**transport entity**）。传输实体可能位于操作系统的内核，或者在一个独立的用户进程中，或者以一个链接库的形式被绑定在具体的网络应用中。网络层、传输层和应用层之间的关系如图10-3所示（其中的**TPDU**为传输协议数据单元）。但是只有资源子网中的终端设备才具有传输层功能，通信子网中的设备一般至多只具备OSI/RM低3层的功能，即通信功能。

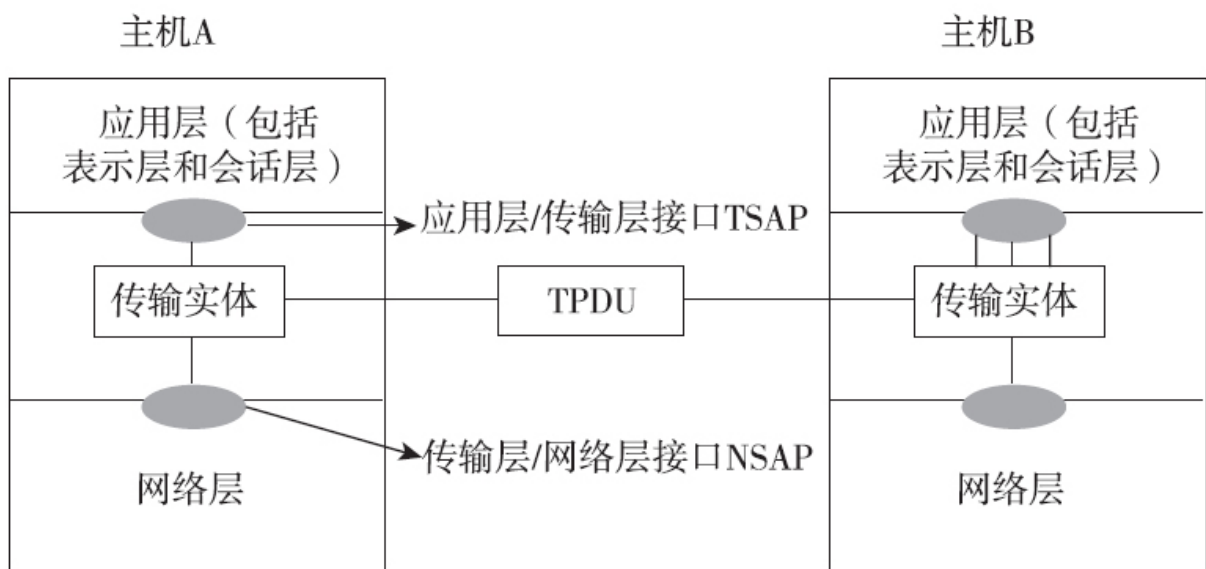


图 10-3 网络层、传输层和应用层的关系

10.1.2 传输层的端到端传输服务

也许有人会说，传输层的主要作用就是为它的上层提供端到端的数据传输服务。那么，什么叫端到端服务呢？其实这里面包含两层含义：第一层含义是从物理的网络连接角度来讲的，即端到端是指网络通信的双方不是在同一链路上，不是点对点连接的；第二层含义是从虚拟的传输连接角度来讲的，即端到端是指在用户看来两端的连接是直接进行的（其实并不是这样的），屏蔽了核心网络结构和各种子网间的差异。当然，这也是体系结构中经常讲的“对等通信”原理。

这里涉及两个概念，即“点对点”（Point-to-Point）连接和“端到端”（end-to-end）连接。所谓“点对点”连接就是通信双方直接通过电缆进行的连接，也就是通常所说的背对背连接，中间没有经过任何其他设备。如图10-4所示，路由器R1与主机A、R2与主机B、R3与主机C这三对之间的连接，以及R1与R2、R1与R3、R2与R3这三对之间的连接都属于点对点连接，因为在这些连接链路的中间没有任何其他设备。而“端到端”连接是两个终端系统之间的连接，体现在两个终端系统的连接中时要经过一个或多个设备。如图10-4所示，主机A与主机B、主机C与主机A、主机B与主机C这三对之间的连接都属于“端到端”连接，它们之间都经过了一个或多个路由器的转连。

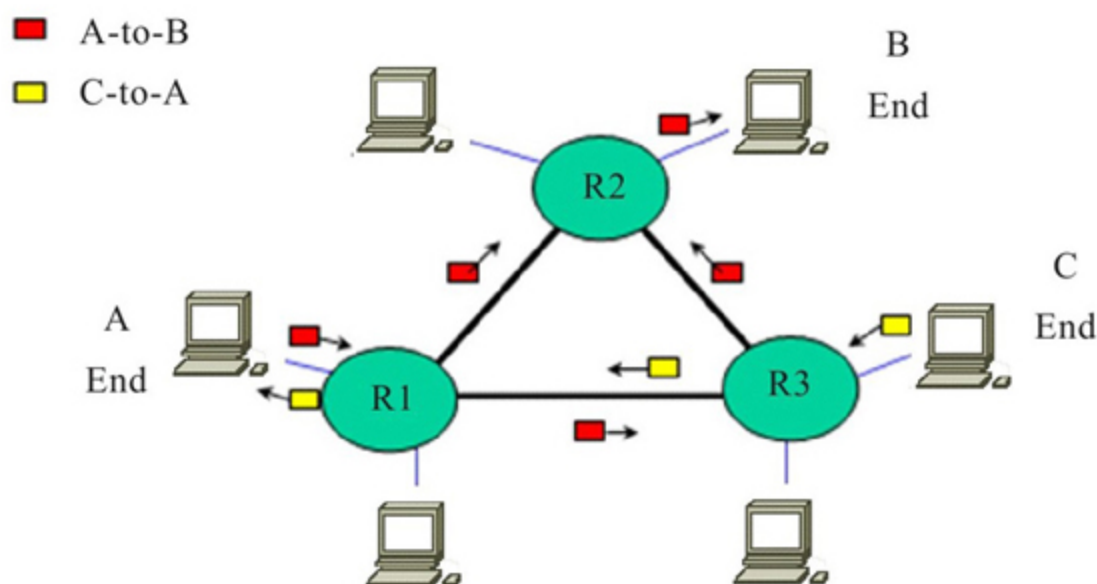


图 10-4 “点对点”连接与“端到端”连接示例

与“点对点”连接和“端到端”连接相对应的是“点对点”传输和“端到端”传输。在进行数据传输前，“端到端”传输需要在两端所经过的线路上建立一条构建于网络层已搭建好的路由路径之上的虚拟传输连接

（当然，这里仅是针对面向连接的传输层协议而讲的），以此来屏蔽所经网络类型或参数配置的不同，就像它们是直接相连的一样；链路建立后，源端就可以发送数据，直至数据发送完毕，接收端确认接收成功。

“端到端”传输的优点是链路建立后源端知道接收设备一定能收到（这就是“可靠性”的体现），而且经过中间交换设备时不需要进行存储转发，无须建立多条传输链路，因此传输延迟小。“端到端”传输的缺点是直到目的端收到数据为止，源端的设备一直要参与传输，因为

只有在传输完成后，从源端到目的端所建立的虚拟传输连接才能“拆除”，否则会中途中断数据的传输。这样一来，如果整个传输的延迟很长，那么对源端的设备会造成很大的浪费，因为那时源端可能早就把所有数据都发完了，只是因为中间子网的一些延迟原因而导致接收端长时间没有接收完数据，最终导致传输连接不能释放。“端到端”传输的另一个缺点是，如果接收设备关机或故障，那么端到端传输不可能实现，因为无法在两端建立一条虚拟的专用传输通道。

“点到点”传输是指源端可直接把数据传给与它直接相连的设备，在需要的时候又可把数据传给与之直接相连的下一台设备，通过一台台直接相连的设备，把数据传到接收端。很显然，这是一种接力传输方式。“点到点”传输的优点是源端设备送出数据后，它的任务已经完成，不需要参与整个传输过程，这样不会浪费源端设备的资源，后面的数据传输是由后面的设备来完成的。另外，即使接收端设备关机或故障，点到点传输也可以采用存储转发技术进行缓冲，当然缓冲也是有大小和时间限制的。“点到点”传输的缺点是源端发出数据后，不知道目的端能否收到或何时能收到数据，因为它不需要事先与对方建立连接，数据发送也是以数据报方式进行的，即不管对方是否工作正常都把数据发送过去。

在一个网络系统的不同层次中，可能用到端到端传输，也可能用到点到点传输。例如在Internet中，网络层及以下各层都采用点到点传

输，传输层以上采用端到端传输。

10.1.3 传输层服务

“传输层”在通信子网提供的服务基础上，为源主机和目的主机之间提供可靠的、透明的、可由用户选择的数据传输服务，是面向用户的高层和面向通信子网的低层之间的软件接口，或者称为“桥梁”。同时，传输层也弥补了高层所要求的可靠、可控制服务与网络层所提供的非可靠服务之间的差距，并向高层屏蔽通信子网中不同网段的差异。

1.传输层服务类型

与网络层有面向连接服务和无连接服务两大类一样（在TCP/IP体系结构中只有无连接的网络层服务），传输层的服务也有这两大类。面向连接的传输服务在提供传输服务前需要先建立专门的传输连接，而且这条连接是可管理的，在需要或通信结束时进行拆除。面向连接的传输服务是可靠的传输服务，而且可提供拥塞控制和差错控制功能，如TCP提供的传输服务。无连接的传输服务在提供服务前不需要建立专门的传输连接，直接向目的节点发送数据，不管是否有可传输的通道，只提供不可靠（仅做尽力传输）的传输服务，如UDP提供的传输服务。

此处所说的“面向连接的传输服务”与在网络层中介绍的“面向连接的网络层服务”十分相似，两者都向用户提供连接的建立、维持和拆除，而且无连接的传输服务与无连接的网络层服务也十分相似。那么，既然传输层服务与网络层服务如此相似，又为什么要将它们划分成两个层次呢？因为网络层是通信子网的一个组成部分，网络服务质量并不可靠，如频繁地丢失分组，网络层系统可能崩溃或不断地进行网络复位。对于这些情况，用户将束手无策，因为用户不能对通信子网加以控制，因而无法采用更优的通信处理机来解决网络服务质量低劣的问题，更不能通过改进数据链路层纠错能力来改善它。解决这一问题的唯一可行方法就是在资源子网的主机上增加一个可以由用户控制的传输层。通过传输层的面向连接服务，使得数据传输更加可靠，特别是对于IP这类无连接网络层服务。

2.服务等级

不同类型的网络，以及不同连接性能的网络，所提供的传输层服务级别也不一样。

传输层提供的服务可分为“传输连接”服务和“数据传输”服务。“传输连接”服务是传输层为它的上层发出的每个应用进程在网络层提供的服务基础之上建立一个相应的逻辑连接；而“数据传输”服务则是在所建立的传输连接基础上，为用户提供比网络层更有保障（主要针对面

向对用户的传输层服务而言）、更有针对性，并且能由用户自己进行拥塞控制、差错控制的数据传输。

传输层所能提供的服务质量的水平，不仅与所选择的传输层协议有关，更重要的是与所服务的网络类型有关，毕竟传输层是建立在网络层基础之上，依赖网络层所提供的路由服务。根据不同网络所提供的不同传输层服务质量水平，可把传输层服务分为以下三大类。

□A类：具有低的连接差错率和故障率，是可靠的传输服务，一般指在虚电路网络上。

□B类：具有较低的连接差错率和故障率，传输服务质量中等，应用于广域网和互联网中的传输层协议多数属于这一类。

□C类：具有较高的连接差错率和故障率，服务质量较差，提供数据报服务或无线电分组交换网所提供的传输层协议均属此类。

3.传输层协议类别

在OSI/RM体系结构的传输层中，不同类型的传输层协议所支持的服务级别不一样，共分TP0~TP4五种类型。其中，TP0~TP3这四种类别均属于面向连接的传输层协议，发送数据前必须先建立传输连接；TP4既可以是面向连接的传输层协议，也可以是无连接的传输层协议。

□TP0：简单类，只建立一个简单的端到端传输连接，提供基本的数据段和重组功能。TP0传输层协议先识别网络层所支持的最大传输单元（MTU）大小，然后据此在必要时对数据包进行分段，在接收端再对数据段进行重组。

□TP1：基本差错恢复类，在TP0基础上新增了差错控制功能，可以对每个TPDU进行编号，当收到不确认（NAK）消息或者有TPDU丢失时将会重发；如果有太多的不确认TPDU，TP1类别的传输层协议还可重新初始化传输连接。

□TP2：多路复用类，在TP0基础上新增了传输连接多路复用和解复用功能，以及流量控制功能。

□TP3：差错恢复和多路复用类，是TP1和TP2传输层协议功能的综合。

□TP4：差错检测和恢复类，在TP3基础上新增差错检测功能，是最复杂的传输层协议类型。在功能上，类似于TCP/IP体系结构中的TCP，但TP4类别的传输层协议既支持面向连接的网络服务，也支持无连接的网络服务。

上述这五种传输层协议类别的综合比较如表10-1所示。

表 10-1 五种传输层协议类别的综合比较

协议类别	主要功能	传输层协议类型
TP0	数据段和重组	面向连接的传输层协议
TP1	数据段和重组、差错控制	
TP2	数据段和重组、多路利用和解复用	
TP3	数据段和重组、差错控制、多路利用和解复用	
TP4	数据段和重组、差错控制、多路利用和解复用、差错检测	既可以是面向连接的传输层协议， 也可以是无连接的传输层协议

10.1.4 TSAP和TPDU

在OSI/RM体系结构的传输层中，涉及两个非常重要的术语——**TSAP**（传输服务访问点）和**TPDU**（传输协议数据单元），它们贯穿整个传输层数据传输服务的始末。下面具体介绍一下这两个术语。

1.TSAP

在计算机网络中，传输层要在用户之间提供可靠、有效的端到端服务，必须把一个用户进程和其他的用户进程区分开，这个工作主要就是由传输层地址来完成了。在本书第5章介绍数据链路层时就讲过，不同层间之间的互访是通过**SAP**（服务访问点）进行的。**SAP**位于除最高的应用层外其他各层的上边缘，用来为下层向上层提供服务，同时是上层调用下层服务的逻辑接口。在传输层的上边缘也有对应的**SAP**（而且可能远不止一个），那就是**TSAP**。**TSAP**是上层调用传输层服务，以及传输层为它的上层提供服务的逻辑接口，是传输层地址，如图10-2所示。

同一时间、同一对网络实体间的用户应用进程可能有多个，不能仅靠网络实体地址（即**NSAP**）来标注通信双方（因为此时通信的实体是各个应用进程，而不是通信双方主机），而必须借助传输层地址进

行标识。TSAP相当于传输层的地址，不同的TSAP标识不同的会话或应用进程。

为确保所有的传输地址在整个网络中是唯一的，因此将传输地址分成网络ID、主机ID及主机分配的端口三部分。前面的“网络ID”和“主机ID”这两部分正好是IP地址的两部分，用来指定具体的网络实体，是NSAP（网络服务访问点）；后面的“端口”是传输层特定的属性，用来与应用进程进行一一对应的，所以说真正的传输层地址其实就是具体应用所占用的“端口”。

值得注意的是，在总共65535个对应传输层协议端口中，前1024个端口被称为常规端口，代表了特定的应用层服务，或者说是应用层进程的端口号，不能分配给其他应用服务使用（其他的端口可以随意使用），而且同一时间、同一主机上不能有两个应用进程使用相同的端口号，因此，在为应用进程分配端口时不能随意。例如，将80号TCP端口分配给了WWW（对应HTTP应用）服务使用，将23号TCP端口分配给了Telnet服务使用，将21号TCP端口分配给了FTP服务使用等。

2.TPDU

在本书第5章同样也介绍过，在分层的网络体系结构中的每一层都是使用PDU（Protocol Data Unit，协议数据单元）来与对方的对等层进行通信的。PDU包含来自上层的信息，以及当前层的实体附加的信

息。然后，这个PDU被传送到下一个较低的层。其实，在其他层中我们都提到了PDU这个术语，只不过到了最后不是直接说是某某PDU，而是为各层起了一个专用的名称，如在数据链路层的LLC和MAC两个子层中所说的“帧”就分别对应了LPDU（逻辑链路协议数据单元）和MPDU（介质访问协议数据单元），另外，网络层中所说的“分组”或“包”其实对应的就是NPDU（网络协议数据单元）。但到了传输层，我们在本书第3章讲过，传输层的数据单元是“数据段”，这个名称显然太容易混淆，于是在OSI/RM体系结构的传输层中还是把其中传输的数据单元称为TPDU（不过，在TCP/IP体系结构中，TCP的协议数据单元仍然为“数据段”）。其实，TPDU与本书前面几章中介绍的“比特”、“帧”“分组”（或“包”）是同类概念。

既然TPDU是传输层的协议数据单元，那么它与数据链路层的“帧”及网络层的“分组”一样，也有它的基本格式，即由“TPDU头部”和“TPDU有效载荷”两部分组成，与其他层中的PDU基本格式一样。其中，“TPDU有效载荷”是来自会话层（OSI/RM体系结构中）或应用层（TCP/IP体系结构中）的数据；而“TPDU头部”由“长度指示”（LI）、“固定部分”、“可变部分”三个基本部分组成。整个TPDU的基本结构如图10-5所示。



图 10-5 TPDU基本结构

□LI: 占1字节, 以字节为单位标识整个TPDU头部的长度, 最大值为254。

□固定部分: 包括TPDU代码和常用参数, 其内容对于某一特定的TPDU来说是固定的, 如包括目的端口和源端口等, 但是不同传输层协议的TPDU是不一样的。

□可变部分: 包含一些可选参数项, 每个参数项由三个字段构成: **Code** (代码, 1字节)、**Length** (长度, 1字节) 和 **Value** (值, 长度可变), 也就是通常所说TLV。因为可变部分的各构成部分所包含的参数项数目不同, 所以其长度是可变的。OSI/RM中定义的常见TPDU类型参数以及它们对应的代码值如表10-2所示。

表 10-2 OSI/RM 中定义的常见 TPDU 类型

TPDU 类型	名 称	适用的传输协议类别					代码	可携带的有效载荷大小
		0	1	2	3	4		
CR	连接请求	√	√	√	√	√	1110	≤ 32 字节
CC	连接确认	√	√	√	√	√	1101	≤ 32 字节
DR	断开连接请求	√	√	√	√	√	1000	≤ 64 字节
DC	断开连接确认		√	√	√	√	1100	无
DT	数据	√	√	√	√	√	1111	协商好的长度以下
AK	数据确认		√	√	√	√	0110	无
ED	加速数据		√	√	√	√	0001	≤ 16 字节
EA	加速数据确认		√	√	√	√	0010	无
RJ	拒绝		√		√		0101	无
ER	错误	√	√	√	√	√	0111	无

常见TPDU类型说明如下。

□CR（Connection Request，连接请求）：向对端发送的面向连接的传输连接请求的TPDU。该类TPDU头部为7字节，整个TPDU长度可变。

□CC（Connection Confirm，连接确认）：向连接请求者发送的面向连接的传输连接确认的TPDU。该类TPDU头部为7字节，整个TPDU长度可变。

□DR（Disconnect Request，断开连接请求）：向对端发送的断开连接请求的TPDU。该类TPDU头部为7字节，整个TPDU长度可变。

□DC（Disconnect Confirm，断开连接确认）：向连接请求者发送的断开连接确认的TPDU。该类TPDU头部为6字节，整个TPDU长度可变。

□DT (Data, 数据)：向对端发送的真正用户数据的TPDU。该类TPDU头部为3字节，整个TPDU长度可变。

□AK (Acknowledgement, 数据确认)：向数据发送者发送的用户数据确认的TPDU，指出希望收到的下一个TPDU号。该类TPDU头部为5字节，整个TPDU长度可变。

□ED (Expedited Data, 加速数据)：向对端发送的加速数据的TPDU。该类TPDU将优先普通用户数据TPDU发送，其中TPDU头部为5字节，整个TPDU长度可变。

□EA (Expedited data Acknowledgement, 加速数据确认)：向加速数据发送者发送的加速数据确认的TPDU。该类TPDU将优先普通用户数据确认TPDU发送，其中TPDU头部为5字节，整个TPDU长度可变。

□RJ (Reject, 拒绝)：向连接请求者发送的拒绝连接请求的TPDU。该类TPDU头部为5字节。

□ER (Error, 错误)：向数据发送者发送的数据错误的TPDU，表示所收到的对应TPDU有错误。该类TPDU头部为5字节，整个TPDU长度可变。

具体来说，TPDU头部的后两部分要对应具体的传输层协议（也就是上文所说的TP0~TP4）才有意义。但无论是哪种TPDU，在发送端都要自上而下传输，到了网络层后同样要经过网络层协议的封装，形成分组，然后继续向下传输到数据链路层，最后封装成帧。TPDU的逐层封装示意图如图10-6所示。

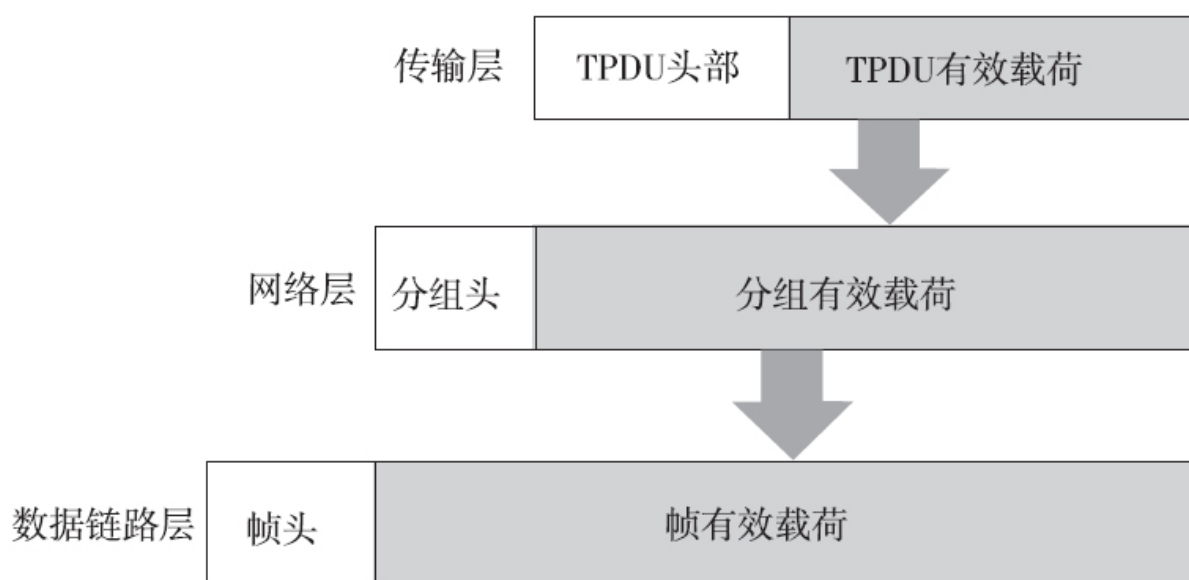


图 10-6 TPDU的逐层封装示意图

10.1.5 传输连接建立阶段的主要TPDU

从本节开始介绍一些主要TPDU格式，看看数据传输的各个阶段使用了哪些主要TPDU，以及它们所包含的基本内容是什么。本节介绍在传输连接建立阶段所用的主要TPDU格式。

从表10-2可知，在传输连接建立阶段，主要用到了CR（连接请求）TPDU和CC（连接确认）TPDU两种。

1.CR TPDU

CR TPDU的基本格式如图10-7所示，下面是各字段的说明（仅说明“固定部分”）。

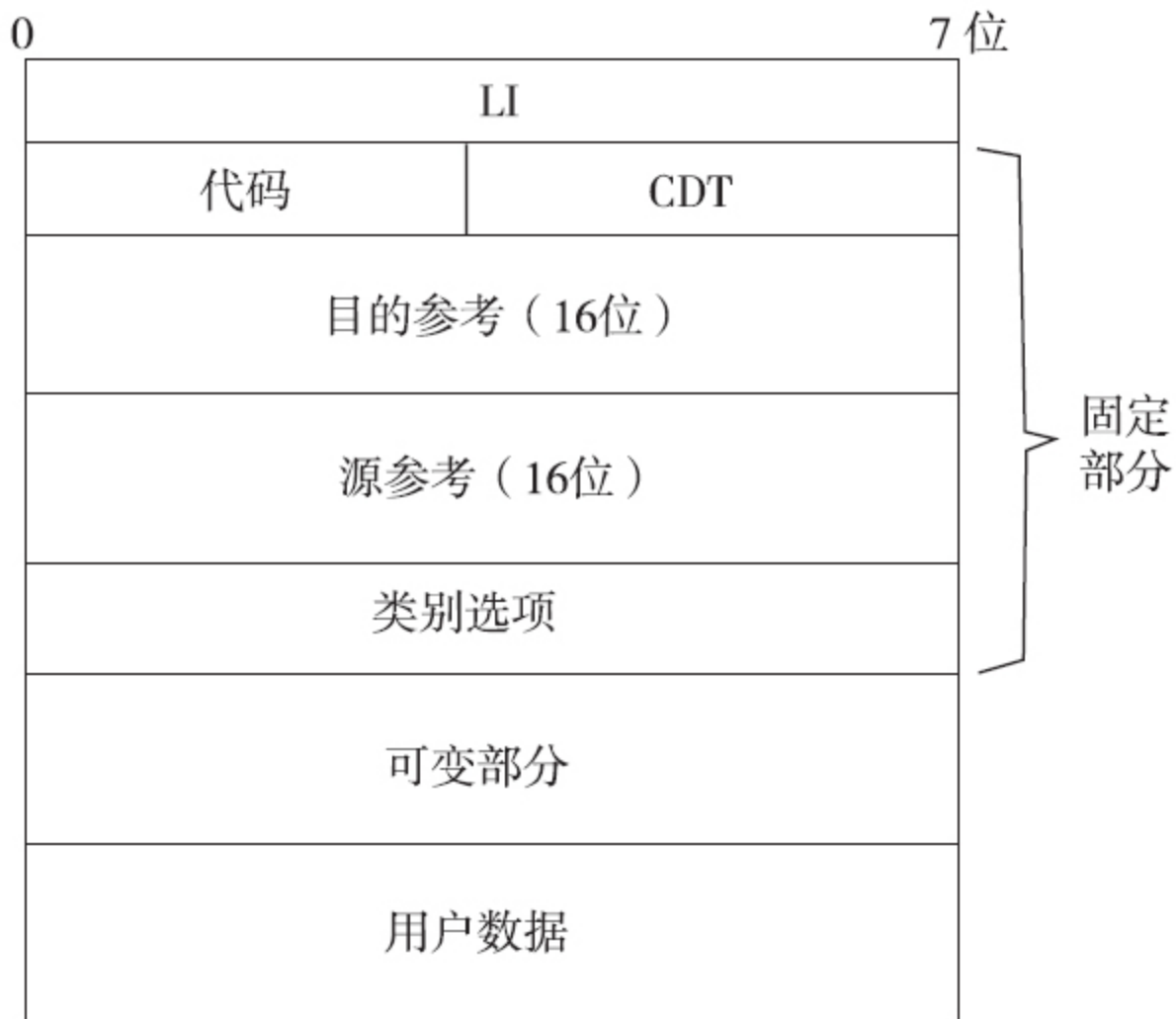


图 10-7 CR TPDU格式

□代码（Code）：标识TPDU类型，占4位，CR TPDU的代码值为1110。

□CDT（Credit Allocation）：信用量分配，表示能接收的TPDU序号的上限值（暗示了可接收的TPDU数量），占4位。CDT类似于本章后面在介绍TCP数据段格式中所说到的“窗口大小”字段，对端通过对

CDT字段值的识别以改变当前的发送窗口大小。TPDU中的CDT字段与TCP数据段的“窗口大小”字段一样均用于流量控制，其基本原理类似于数据链路层和网络层的滑动窗口式流量控制。CR TPDU中的CDT属于初始信用量，在传输连接建立时双方协商确定。

□目的参考（DST Reference）：指该传输连接的目的端对等传输实体用来唯一标识这一连接的标识符，也就是对端传输连接序号，占16位。CR TPDU中的“目的参考”为全0，因为连接尚未建立（正在请求中），将由目的端选择。

□源参考（Source Reference）：本地传输连接标识符，也就是本地传输连接的序列号，占16位。

说明 在传输连接的每一端，源TSAP地址和目的TSAP地址写在CR TPDU头部的可变部分中。当目的端要对收到的CR TPDU表示确认，并同意建立传输连接时，就会发回CC TPDU，其中也包含了“源参考”和“目的参考”这两个字段值。但此时CC TPDU中的“源参考”和“目的参考”都是针对发送CC TPDU的端点而言的，所以其中的“源参考”也即为在CR TPDU中以全0格式填充的“目的参考”。通过这样两次握手过程，双方就知道了各自的传输连接序号了。

□类别选项：占8位，高4位标识使用的传输层协议类型（TP0~TP4）；低4位是可选项，用来说明在DATA TPDU和AK TPDU

中的序号字段是正常的，还是扩展的，正常为0000，扩展为0010。通常TPDU序号只有7位，但对于TP2~TP4传输层协议来说，序号可以扩展到32位。

协议和序号类型都是可协商的，但在协商过程中，服务质量只能降低而不允许提高。协商后的数值通过CC TPDU返回到源端。例如，CR的发送端选择了TP4（值为1000），CC的发送端可以将它降低为TP2（0010）。但若CR的发送端选择的是TP2，则CC的发送端不能将它改为TP4。

CR TPDU头部可变部分可包括TPDU最大长度、源TSAP ID、目的TSAP ID、安全参数、校验和、QoS参数等，具体内容不作介绍。

2.CC TPDU

CC TPDU的格式与CR TPDU类似（参见图10-7），不同的只是CC TPDU中的“代码”字段为1101，代表为CC TPDU，同时“目的参考”字段不再是16位全0，而是发起连接端的传输连接序列号，即CR TPDU中的“源参考号”，它是提供服务的传输连接序号。

10.1.6 数据传输阶段的主要TPDU

从表10-2可知，在传输连接建立阶段，主要用到了DT（数据）TPDU、AK（数据确认）TPDU、ED（加速数据）TPDU和EA（加速数据确认）TPDU四种。这里要特别注意的一点是，在所有TPDU中，只有DT TPDU和ED TPDU才有“TPDU序列号”字段，也就是只有这两种TPDU才会进行编号，其他TPDU均不编号，另外，只有AK TPDU和EA TPDU才有“你的序列号”字段，其他TPDU均没有该字段。

1.DT TPDU

DT TPDU是数据传输的TPDU。在DT TPDU中，选择使用的传输层协议类型不同，对应的格式也不一样。当选择TP0和TP1类型传输协议时，它的格式如图10-8所示。各字段说明如下（仅说明“固定部分”）。

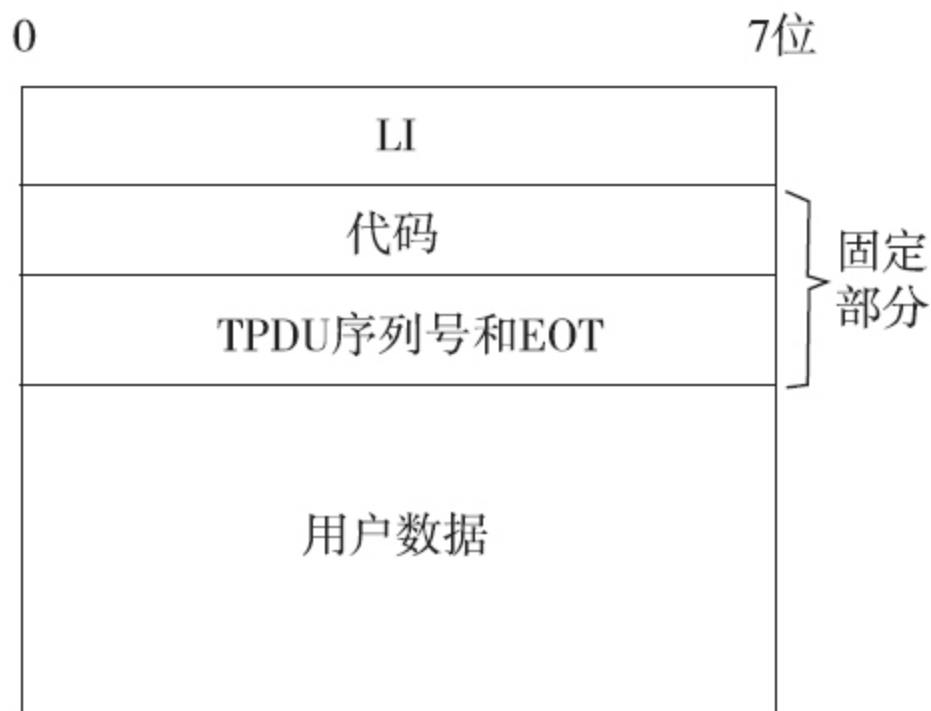


图 10-8 TP0和TP1类型的DT TPDU格式

□代码：标识TPDU类型，占8位，DT TPDU的代码值为1111 0000。

□TPDU序列号和EOT：占8位，高7位标识对应DT TPDU的序列号；最低位（EOT位）用来标识此TPDU是否是某个数据的最后一个数据段，等于1时表示是最后一个数据段。

当选择TP2、TP3和TP4类型传输协议时，它的格式要分两种情况：一是在建立传输连接时选择的不是TP2~TP4类型，此时的TP2、TP3和TP4类型DT TPDU格式如图10-9所示；二是如果在建立传输连接

时也选择的是TP2~TP4类型，此时的TP2、TP3和TP4类型DT TPDU格式如图10-10所示。



图 10-9 建立连接时未选择TP2~TP4类型时的TP2~TP4 DT TPDU 格式

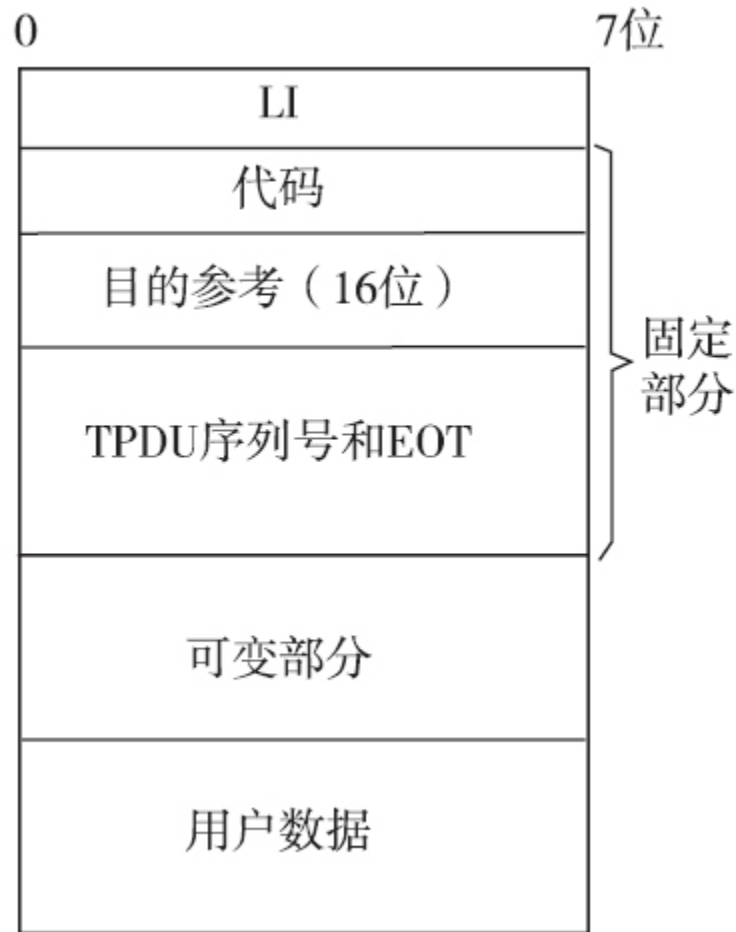


图 10-10 建立连接时已选择TP2~TP4类型时的TP2~TP4 DT TPDU 格式

在以上两种情况下，DT TPDU格式的主要区别在于“TPDU序列号和EOT”字段的长度，图10-9中的为8位，而图10-10中的为16位。对比图10-8可以看出，在图10-9和图10-10所示的格式中，主要是多了一个“目的参考”和“可变部分”。因为TP2~TP4类传输层协议支持多路复用，需要标识各传输连接号，所以需要“目的参考”这个字段，而TP0~TP1类型传输层协议是不支持多路传输连接复用的，所以即使在

传输数据时不标识传输连接号，也不会出现差错。“目的参考”字段占16位，是在通过CR TPDU和CC TPDU建立传输连接过程中获得的目的端传输连接号。在“可变部分”中包括“校验和”参数。

2.AK TPDU

AK TPDU是接收端在收到源端发来的DT TPDU后的确认TPDU。它的TPDU格式分为在建立传输连接时没有选择具体的协议类型时的“正常”格式和在建立传输连接时已选择TP1~TP4类型传输协议时的“扩展”格式，分别如图10-11和图10-12所示。两者的主要区别为“代码”、“CDT”和“你的TPDU序列号”3个字段的宽度。下面分别介绍其中的各个字段（仅说明“固定部分”）。

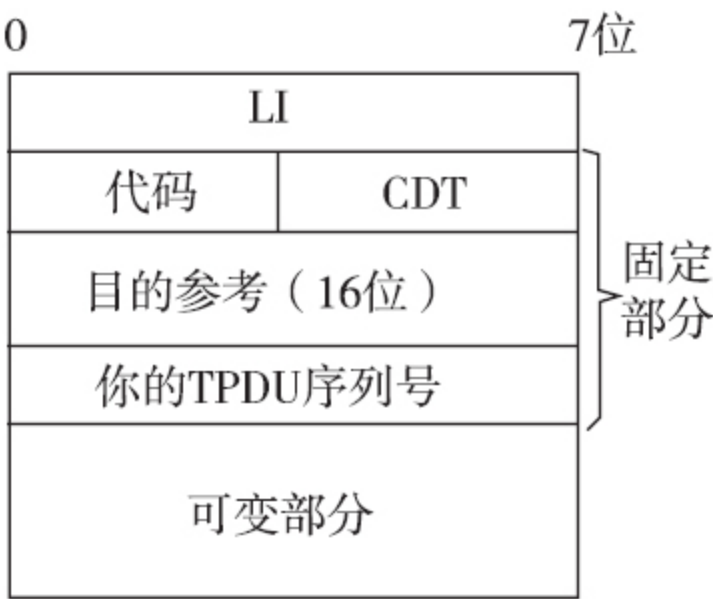


图 10-11 正常情况下的AK TPDU格式

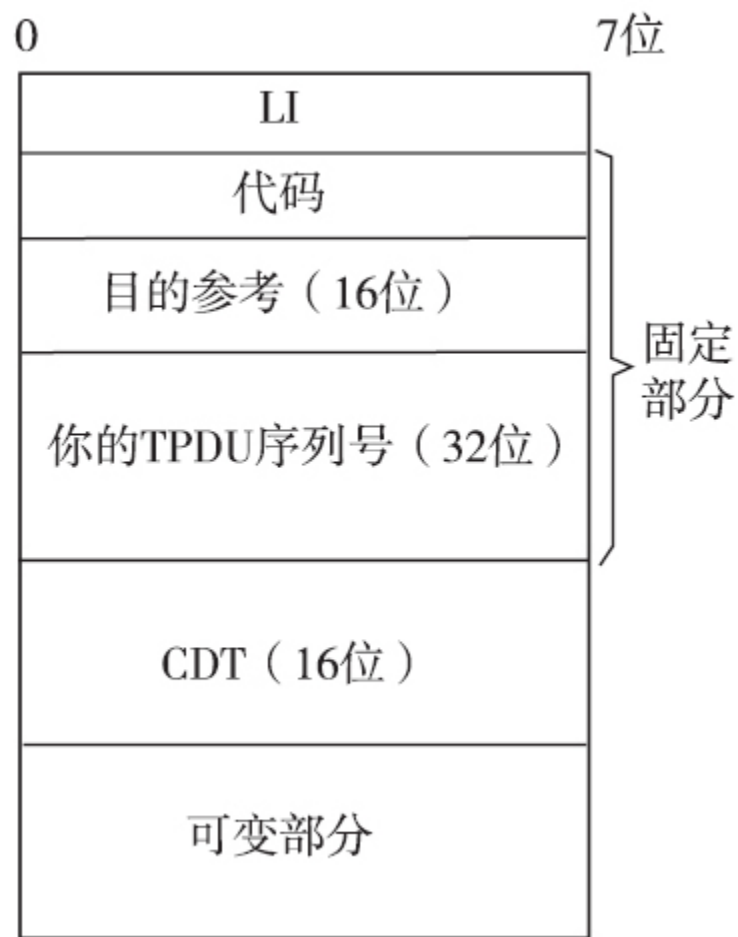


图 10-12 扩展情况下的AK TPDU格式

□代码：标识TPDU类型，在正常情况下占4位，为0110，而在扩展情况下占8位，为0110 0000，代表AK TPDU。

□CDT：信用量分配，用于流量控制，在正常情况下占4位，而在扩展情况下占16位。

□目的参考：占16位，是数据传输源端的传输连接序号。

□你的TPDU序列号（YR-TU-NR, Your-TPDU-Number）：表明所期望接收的下一个DT TPDU的序号，这也隐含地表示对该序号以前的各DT TPDU均已加以确认，相当于本章后面将要介绍的TCP数据段中“确认号”字段。在正常情况下占8位，而在扩展情况下占32位。

为了避免因“YR-TU-NR”字段位数不足而出现序混乱现象，还可以利用AK TPDU中的“可变部分”参数的“子序号”的参数（还包括“校验和”和“流量控制确认”参数）进行TPDU序号标识。子序号的使用规则如下：

□当给一个具有更高序号（也就是“YR-TU-NR”字段值还未达到最大值）的DTTPDU发回AK TPDU时，应将“子序号”复位置0，表示不采用“子序号”字段。

□若本次发送的AK TPDU的“YR-TU-NR”字段值不变，只是信用量CDT字段值增大了，则“子序号”不变，即与上次发送的AK TPDU的“子序号”值相同。

□若本次发送的AK TPDU的“YR-TU-NR”字段值不变，但信用量CDT字段值减小了，则需要在上次发送的AK TPDU的“子序号”值基础上加1。

在接收AK TPDU的一端，仅在以下情况时才将AK TPDU收下：

□它所确认的DT TPDU的序号比上次收到的高。

□它所确认的DT TPDU的序号未变，但具有一个更高的子序号。

□它所确认的DT TPDU的序号未变，所携带的子序号也未变，但具有一个更大的信用量CDT。

若同时收到若干个AK TPDU，则按YR-TU-NR字段从小到大进行处理。当“YR-TU-NR”字段值相同时，按“子序号”字段值从小到大进行处理。如果“子序号”字段值也相同，则按信用量CDT字段值从小到大进行处理。

3.ED TPDU

ED TPDU只能选择TP1~TP4类型传输协议进行传输，它与DT TPDU一样也分为在建立连接时选择了TP2~TP4传输类型和未选择TP2~TP4传输类型两种情况，对应的格式也与DT TPDU的两种情况下的格式完全一样（分别参见图10-9和图10-10），不同的只是其中的“代码”值为0001（建立传输连接时未选择TP2~TP4协议类型时）或0001 0000（建立传输连接时已选择TP2~TP4协议类型时）。

对ED TPDU的流程控制比较简单，因为在任何时刻仅允许一个未被确认的ED TPDU在通信途中，也就是说，发送窗口大小为1。在接收端，即使当正常数据的信用量的大小为0时也必须接受收到的ED

TPDU。当收到重复的ED时也要发回确认，不过要将重复的ED TPDU 丢弃。

一个ED TPDU应该保证不能比在它以后发出的正常数据DT TPDU 更迟地交付给接收端的传输用户。为了做到这一点，TP4规定传输实体将传输用户发过来的并已收到的正常DT TPDU暂停发送，而先发送加速数据。只有在加速数据确认（EA TPDU）收到后才能再发送正常数据。

4.EA TPDU

EA TPDU也只能选择TP1~TP4类型传输协议进行传输，它与AK TPDU一样也分正常和扩展两种情况，对应的格式也与AK TPDU的两种情况下的格式基本一样，分别如图10-13和图10-14所示。对比图10-11和图10-12可见，EA TPDU中只是少了CDT字段（因为加速数据的发送是不按TPDU顺序号发送的，有加速数据就立即发送，是数据传输过程中的少数特例），同时，“代码”字段均为0010 0000，代表是EA TPDU。扩展EA TPDU格式中也没有“序列号”和“EOT”这两部分，这也是因为加速数据是不需要根据TPDU序列号来发送的。其他字段与AK TPDU中的对应字段作用相同。

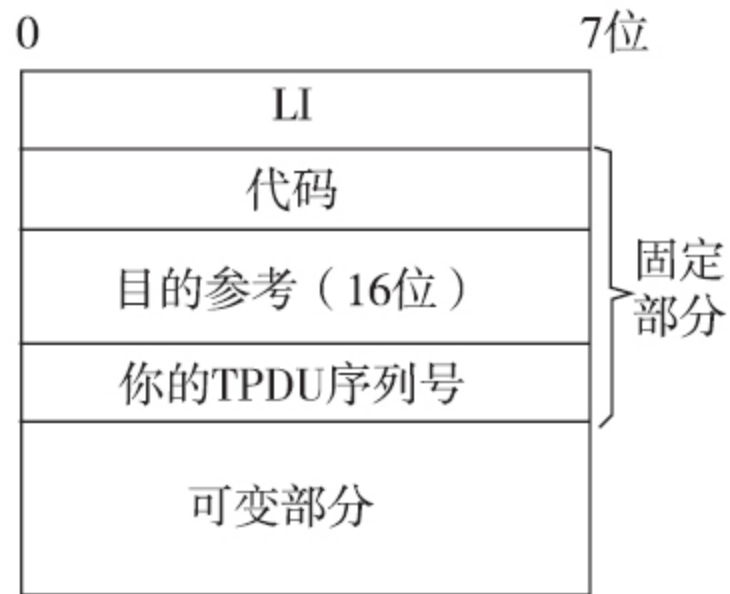


图 10-13 正常情况下的EA TPDU格式

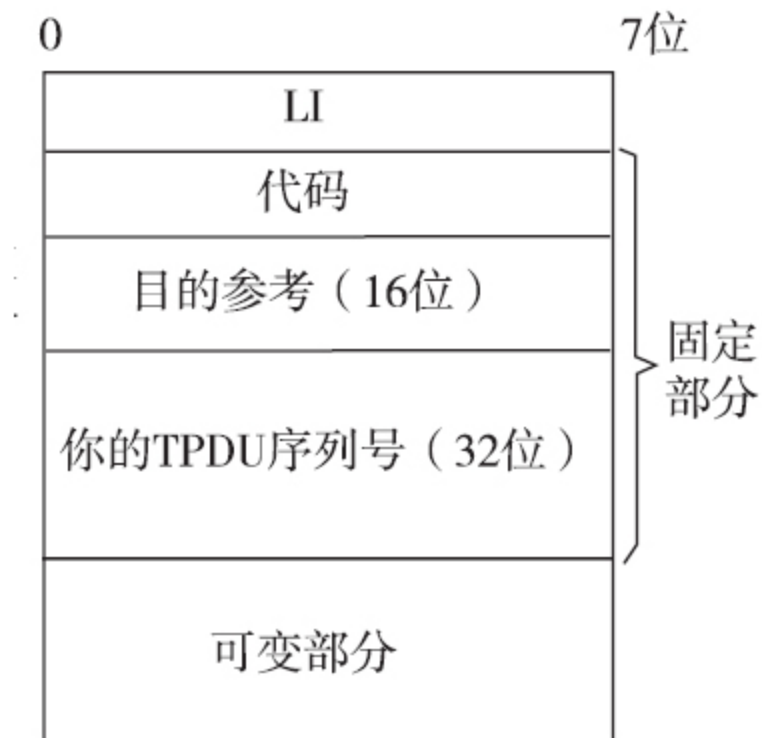


图 10-14 扩展情况下的EA TPDU格式

说明 数据发送的初始信用量分配规则：对于流向目的端的数据，其初始信用量分配由CC TPDU提供；而对于流向源端的数据，其初始信用量分配是由CR TPDU提供的。在以后的数据发送中，信用量分配的大小是通过AK TPDU来分配的。

10.1.7 传输连接释放阶段的TPDU

在连接释放阶段主要的TPDU分为DR TPDU和DC TPDU两种，前者是断开连接请求的TPDU，后者是断开连接确认的TPDU。

DR TPDU和DC TPDU的格式分别如图10-15和图10-16所示。其中DR TPDU“代码”字段的值为1000 0000，DC TPDU“代码”字段的值为1100 0000。DR TPDU中的“原因”字段会显示断开连接的原因，其他字段参见前面对应小节的说明。

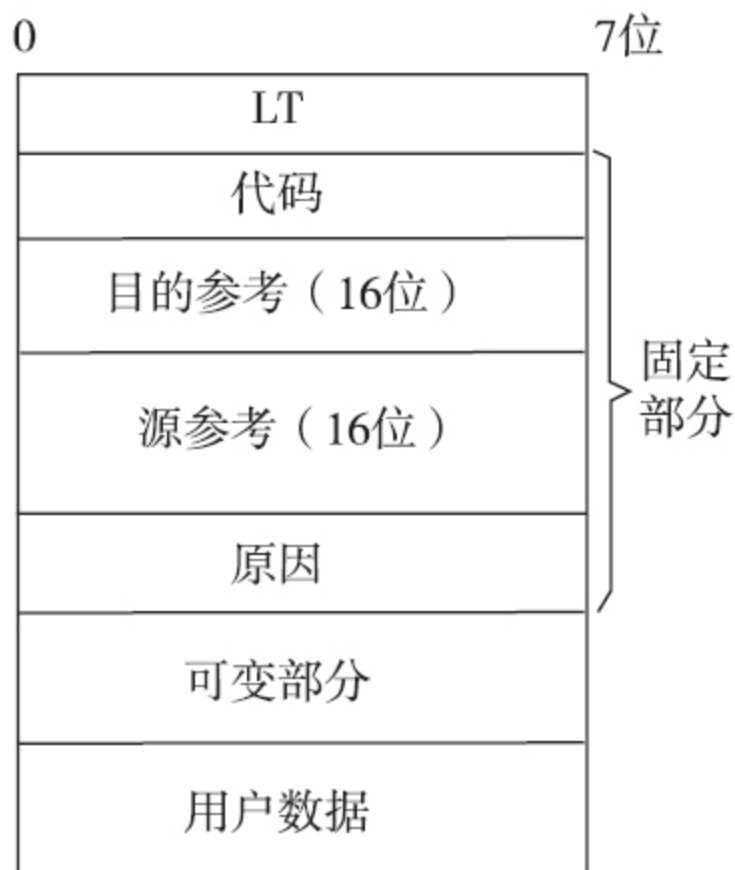


图 10-15 DR TPDU格式

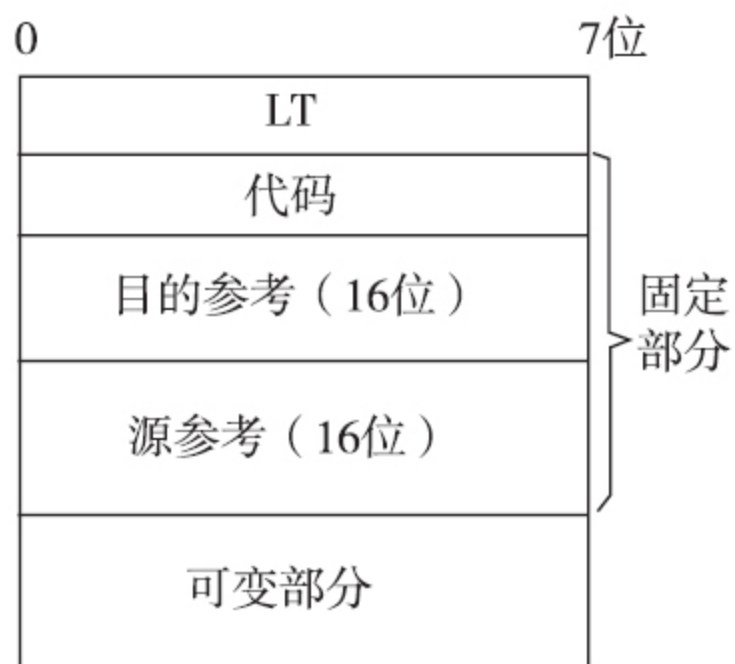


图 10-16 DC TPDU格式

10.1.8 传输服务原语

“服务原语”（service primitive）是服务功能实现的具体过程描述，也就是要完成一项服务，必须要经过哪些主要步骤。一个服务通常由一组原语操作来描述，用户进程通过这些原语操作可以访问该服务。这些原语告诉该服务执行某个动作，或者将某个实体所执行的动作报告给用户。如果协议栈位于操作系统中（大多数情况是这样的），则这些服务原语通常是一些系统调用。这些系统调用会进入到内核模式，然后在内核模式中控制该机器，让操作系统发送必要的分组。到底哪些原语可以使用则取决于所提供的服务，针对面向连接的服务的原语与针对无连接的服务的原语是不同的。在OSI/RM体系结构中，服务原语每层都有，并不只是传输层才有。

传输服务类似于网络服务，但两者存在显著的区别。网络服务倾向于将实际网络提供的服务模型化，由于网络内部问题，实际的网络可能会丢失数据分组，所以网络服务并不十分可靠。相反，面向连接的传输服务则是可靠的，虽然实际的传输服务并非毫无错误，但在不可靠的网络服务之上提供可靠的传输服务正是传输层要实现的目标。

1. 一个简单的传输服务原语模型

在正式介绍传输层服务原语前，先来通过一个最基本、最简单的传输服务原语模型来理解传输服务原语的含义。这个简单的传输服务模型仅包括：监听（Listen）、连接（Connect）、发送（Send）、接收（Receive）、断开连接（Disconnect）。它们代表了一个C/S模式的数据传输服务基本过程，这也是面向连接的传输层服务的5个基本过程。这5个基本服务原语所发送的TPDU和代表的含义如表10-3所示。

表 10-3 5 种基本的传输服务原语

服务原语	可发送的 TPDU	具体含义
Listen	无	处于阻塞状态，直到有某个进程试图与它建立连接为止
Connect	CR	主动请求建立一个连接
Send	DT	发送用户数据 DT TPDU
Receive	无	处于阻塞状态，直到收到一个 DT TPDU 为止
Disconnect	DR	请求释放一个已建立的连接

现假设一台服务器将被多个客户端访问，这些基本传输服务原语的应用流程如下（从最初状态开始）。图10-17给出了全用这些简单传输服务原语来建立和释放连接的状态图。每一个状态的出现都是一种事件触发，或者是由本地的传输用户执行了一个原语，或者是接收到了一个TPDU分组。不过要注意，此处为了简单起见，假设了每一个TPDU都是单独确认的，而且也采用了对称释放模型，由客户先释放连接。图10-17中实线箭头显示了客户端的状态序列，而虚线箭头则显示了服务器的状态序列。

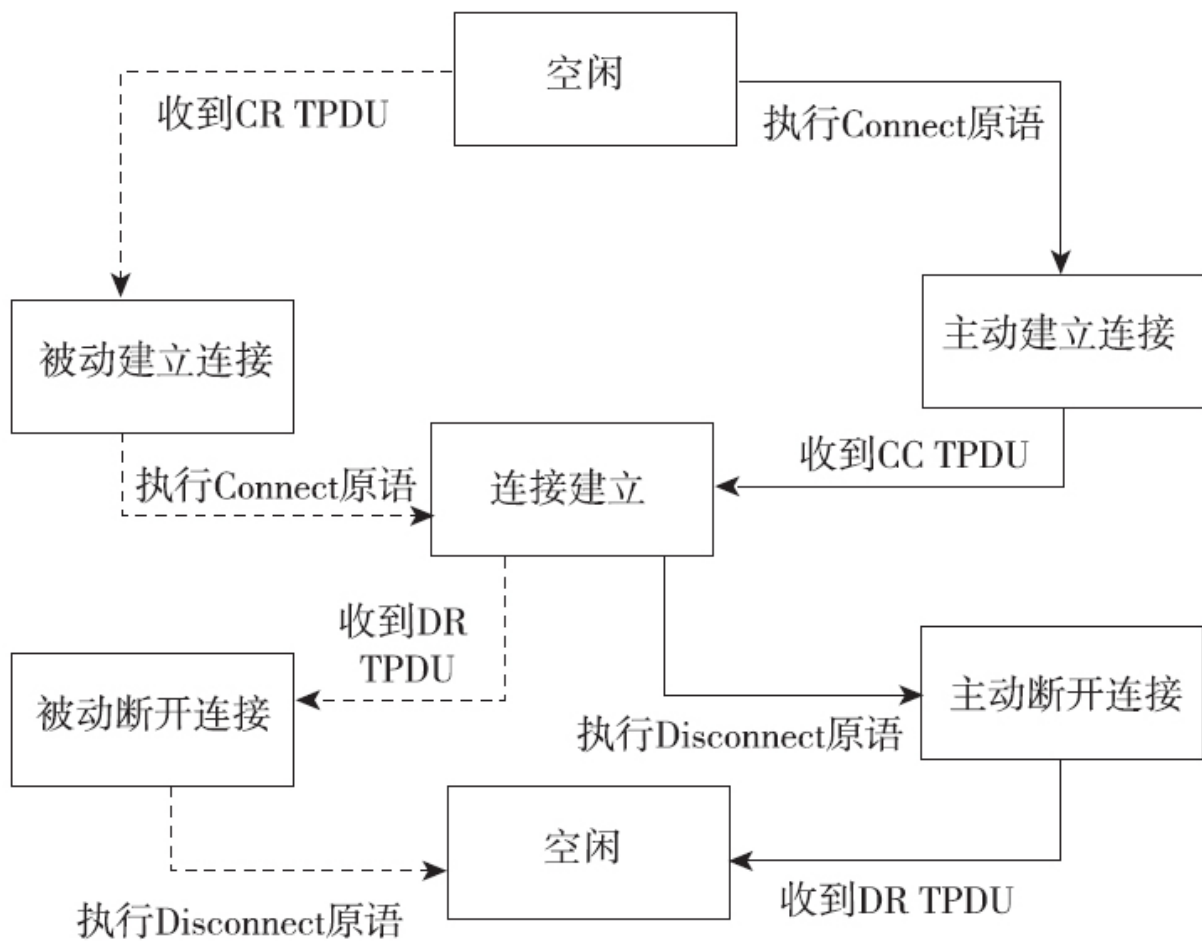


图 10-17 一个简单传输服务原语的应用示例

- 1) 一开始，服务器调用Listen原语，使其处于监听状态，直到有客户来请求连接。
- 2) 当一个客户希望与该服务器进行会话时，它调用Connect原语，向服务器发送一个请求建立传输连接的CR TPDU。
- 3) 当该CR TPDU到达服务器时，服务器传输服务实体执行状态检查，看服务器是否正处于监听状态。如果是，则解除监听，并向客户

端返回一条确认连接请求的CC TPDU。

4) 在这个CC TPDU到达客户端后，客户端获知服务器同意建立传输连接了，随后便通知客户端应用进程开始建立传输连接。

5) 传输连接建议好后，就可以正式发送DT TPDU了。任何一方都可以调用Receive原语，以等待另一方调用Send原语，发送数据。当一方接收到另一方发来的DT TPDU时，接收端会发送一个数据接收确认的DA TPDU。

6) 当不再需要一个连接时，传输用户必须将它释放，以便使两个传输实体内部的缓存空间有机会被重新使用。

释放连接有两种形式：对称释放和非对称释放。在非对称释放中，任何一方都可以调用Disconnect原语，在本端传输实体向对端发送一个DR TPDU后双向传输连接都将被释放。在对称释放中，连接的两个方向彼此独立，每个方向需要单独被释放。当一方调用Disconnect原语时，意味着它不再需要发送数据了，但它仍然希望能接收对方发过来的数据。在这种方式中，只有在双方都调用了Disconnect原语，各自向对方发送DR TPDU后一个传输连接才可能被真正释放出来。

2.ISO规范中的传输服务原语

在OSI/RM体系结构的ISO规范中，各层服务原语分为以下4种类型：

□请求（request）：用户利用它要求服务提供者提供某项服务，如建立连接或发送数据等。

□指示（indication）：服务提供者执行请求原语后，用指示原语通知本端用户实体，告知有人想要与之建立连接或发送数据等。

□响应（response）：收到指示原语后，利用响应原语向对方做出反应，如同意或不同意建立连接等。

□确认（confirm）：请求方可以通过接收确认原语来获悉对方是否同意接受请求。

在这4个服务原语中，request和response由请求服务用户调用，而indication和confirm则是由服务提供者调用。各类服务原语均可以携带参数，如请求原语的参数可能指明它要与哪台机器连接，需要什么服务类别等；指示原语的参数可能包含呼叫者的标识、需要服务的类别等；响应原语的参数可能包括同意或不同意连接的关键字；在确认原语中，则可能对某些参数给出协商值，比如最大数据吞吐量等。

在OSI/RM体系结构的传输层中，主要包括10种传输服务原语，分布在传输连接建立、数据发送和传输连接断开这3个主要阶段中。其

中，在传输连接建立阶段所需的4个服务原语说明如下（小圆点后面部分代表对应的传输服务原语类型，下同）。

□T_CONNECT.request: 传输连接请求原语。客户端通过CR TPDU向服务器请求，让对方建立一条传输连接。

□T_CONNECT.indication: 传输连接指示原语。服务器向自己的用户实体（用户应用进程）指示有客户要求与自己建立传输连接。

□T_CONNECT.response: 传输连接响应原语。服务器通过CC TPDU向客户端发送一条传输连接建立响应。

□T_CONNECT.confirm: 传输连接确认原语。客户端收到服务器的传输连接响应后，向本端用户实体（用户应用进程）确认服务器是否已接受连接请求。

在数据传送阶段所需的服务原语如下。

□T_DATA.request: 数据传输请求原语。向对方提出发送数据请求。

□T_DATA.indication: 数据传输指示原语。向本端用户实体指示有人要向自己发送数据。

□T_EXPEDITED_DATA.request: 加速数据传输请求原语。向对方提出发送回还数据请求。

□T_EXPEDITED_DATA.indication: 加速数据传输指示原语。向本端用户实体指示有人要向自己发送加速数据。

在传输连接释放阶段所需的服务原语如下。

□T_DISCONNECT.request: 释放传输连接请求原语。向对方提供释放传输连接请求。

□T_DISCONNECT.indication: 释放传输连接指示原语。向本端用户实体指示对端要释放传输连接，以便做出相应的处理。

以上10个传输服务原语的综合比较如表10-4所示。

表 10-4 10 个主要传输服务原语综合比较

阶 段	服 务	使用的服务原语	参 数
连接建立	建立传输连接	T_CONNECT.request	被叫地址、主叫地址、加速数据选择、服务质量、传输服务用户数据
		T_CONNECT.indication	
		T_CONNECT.response	服务质量、响应地址、加速数据选择、传输服务用户数据
		T_CONNECT.confirm	
数据传送	正常数据传送	T_DATA.request	传送服务用户数据
		T_DATA.indication	
	加速数据传送	T_EXPEDITED_DATA.request	传送服务用户数据
		T_EXPEDITED_DATA.indication	
连接释放	释放传输连接	T_DISCONNECT.request	传送服务用户数据
		T_DISCONNECT.indication	拆除连接原因、传送服务用户数据

表10-4中，“主叫地址”参数是指请求传输连接的TSAP地址；“被叫地址”参数是指与之建立传输连接的TSAP地址；“响应地址”参数等于被叫地址参数；“加速数据选择”参数是指在传输连接上是否可使用加速数据服务（如果声明没有这项服务，它就不能在传输连接上使用）；“传输服务用户数据”参数是在传输用户之间传送的传输服务用户数据，传输服务提供者不进行修改，其长度为1~32字节；“服务质量”参数包括传输服务用户的一些要求，即QoS参数。

无连接的传输服务只有两个原语，如表10-5所示。

表 10-5 无连接传输服务原语

原 语	参 数
T_UNIT_DATA.request	主叫地址、被叫地址、服务质量、传送服务用户数据
T_UNIT_DATA.indication	主叫地址、被叫地址、服务质量、传送服务用户数据

可用如图10-18所示的状态转移图来说明OSI/RM体系结构中定义的面向连接传输服务在连接建立和连接释放两个阶段的服务原语的使用情况。状态1是空闲状态，表示没有传输连接。每次连接的建立都从这一状态开始，而在连接释放后总是再回到这个空闲状态。状态2和状态3都属于连接建立阶段。状态4是数据传送就绪阶段，实际上也就进入了数据传输阶段，调用相应的原语后就可以进行数据传输了。无论在连接建立阶段还是在数据传送阶段，都可以通过连接释放原语返回到原始的空闲状态。图10-18的状态转移示意图指示在一个传输连接的一个端点上可能发生的状态序列，箭头旁边注明的是对应状态转移时所

需调用的相关原语，在这些原语的交互作用下，状态沿箭头转移到下一个状态。

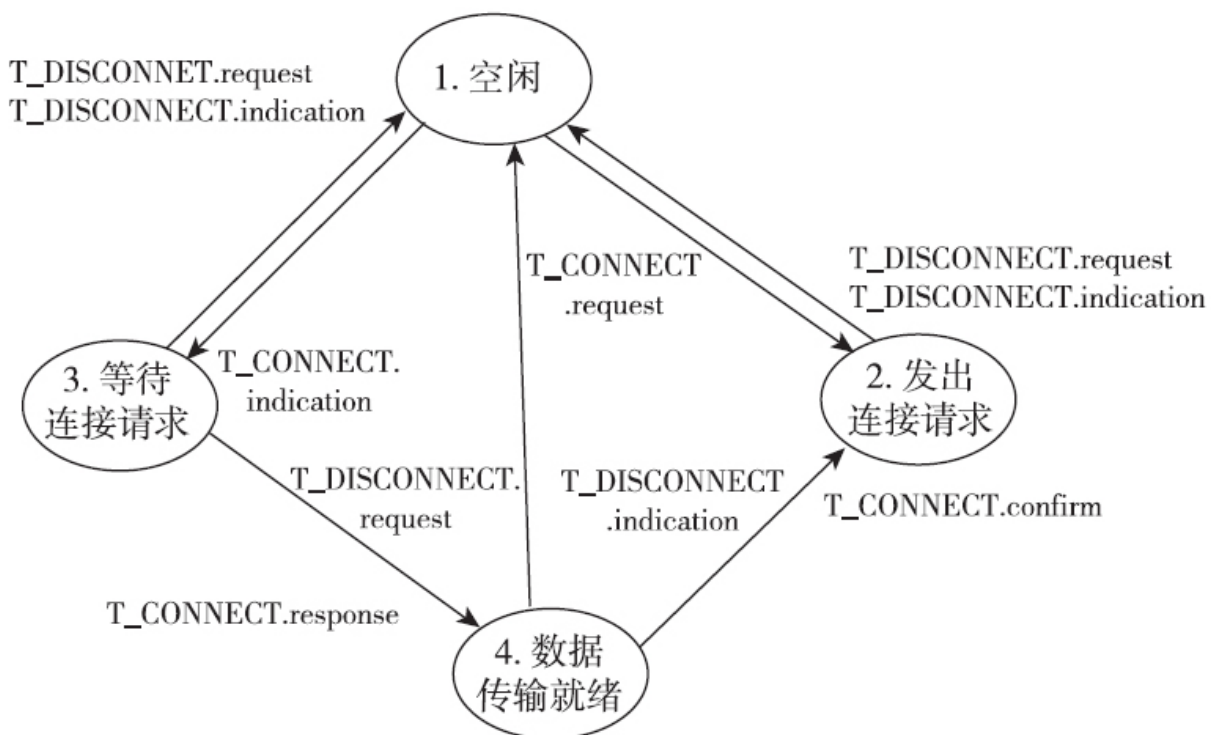


图 10-18 传输连接建立和释放阶段的一端主机状态转移示意图

10.2 传输层服务功能

从上文可知，传输层要完成端到端的透明数据传输，至少要分为两个步骤：一是虚拟传输连接的建立（此处仍仅针对面向连接的传输层协议），二是源端用户数据沿着传输连接传送到目的端。在整个数据传输过程中，传输层服务需要完成以下8个方面的基本功能：传输层寻址、传输连接建立、数据传输、传输连接释放、流量控制、拥塞控制、多路复用和解复用、崩溃恢复。这些是本章的重点与难点，希望读者着重理解这些功能的实现原理。

10.2.1 传输层寻址方案

在本章前面就已说到，虽然网络层把数据分组从源主机传送到了目的主机，但是这并不代表数据的传输过程就全部完成了，因为数据分组还没有真正地交付给相应的应用进程，此时应用进程才是数据通信主体。而我们又知道，同一时间的通信双方可能存在许多并发应用进程，那么源端来的数据分组该交给哪个应用进程呢？这就涉及传输层的寻址问题了。在本章前面已介绍了TSAP（传输服务访问点），它代表的是一个传输层端口，因为传输层是以一个端对应一个应用进程来表示的。

应用进程（包括客户端和服务端）可以将自己关联到一个TSAP上，以便与远程的TSAP建立连接。但是我们知道，只有少数常见的服务（如WWW、FTP、POP、SMTP、DNS、DHCP等）是固定分配了TSAP地址——端口的，绝大多数服务并没有与固定的端口进行绑定，所以这些服务的发送端就需要接收端对应用户进程关联的TSAP地址。那么，发送主机的用户进程如何知道接收主机对应用户进程关联的TSAP地址呢？下文将进行介绍。另外，在一台服务器上有大量潜在的服务进程，但是大多数服务进程又很少使用，那么让每个服务进程都主动、持续地监听一个TSAP地址显然是非常不明智的，也浪费了大量的服务器资源。

1.进程服务器的TSAP地址分配方案

为了解决以上问题，提出了一种更好的方案，那就是“进程服务器”方案，它是“初始连接协议”（initial connection protocol）方案的一种。它先由进程服务器为各应用进程集中建立初始传输连接，然后再转给对应的服务进行正式的传输连接。进程服务器的TSAP地址分配方案的做法不是让每一个对外提供的服务都在一个特定的端口上监听，而是使用一个特殊的进程服务器（process server）来集中为那些较少使用的服务提供监听代理功能。它可以同时监听一组端口，以等待外来的连接请求。每个服务的潜在用户总是从一个Connect请求开始，在

Connect请求中指定了这些用户想要的服务的TSAP地址。进程服务器为用户进程分配TSAP地址的基本流程如图10-19所示，具体解释如下。

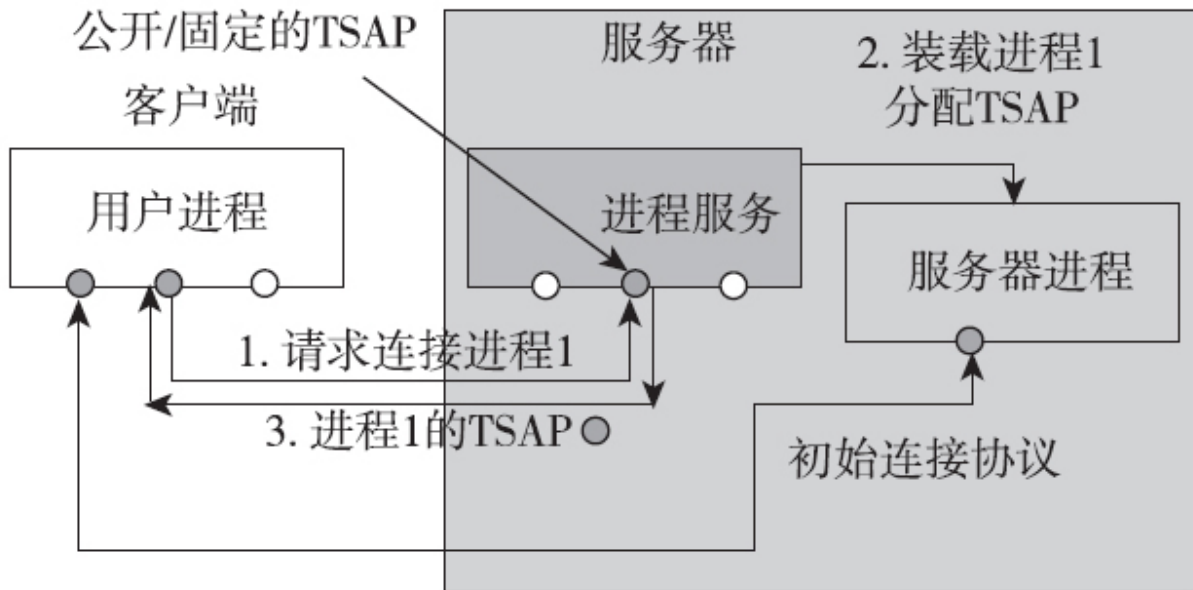


图 10-19 进程服务器为用户进程分配TSAP地址的基本流程

1) 用户向服务器发送一个传输连接请求，如果没有专门的服务器在等待，则会得到一个与“进程服务器”之间的连接。

2) 进程服务器接到了用户的传输连接请求后，装载对应的用户进程，启动与该请求对应的服务。

3) 进程服务器为该用户进程分配一个临时的TSAP地址，这样用户与对应的服务器进行初始连接，允许对应的服务继承进程服务器与用户之间已有的连接，然后对应的服务执行用户请求的工作，而进程服务器则继续监听新的请求。

例如，某公司有多个产品营销部门，每个部门设立了一个专门的售后服务热线电话。但其中有些部门的热线电话比较忙，经常有用户打电话咨询或投诉，而有些部门的热线电话打进来的人比较少。这时，如果仍为每个部门专门安排一个人负责热线电话，显然对于那些用户参与比较少的热线电话来说是一种浪费。这时我们肯定会想到，把这些用户参与比较少的部门的热线电话用一个总线电话来转接，只需要一个专门的接线员来管理总机即可。当有人打这个总线电话时，总机接线员询问所咨询的内容，然后再把电话转接到对应的部门分机，由对应的部门接线员进行处理，总机接线员又继续等待下一个用户电话，而这些部门的热线电话接线员平时可以进行其他工作，不必总在那里等待用户的来电，节省了人力资源。

2.名称服务器的TSAP地址分配方案

虽然以上介绍的“初始连接协议”在大多数情况下都可以满足需求，也可以工作得很好，但是在一些特殊情况下还是有些服务的运行必须独立于进程服务器。如一个文件服务器需要在特殊的硬件上运行，不能当用户要与它通话的时候才临时创建连接。就像前面所说的那些用户参与比较多的部门热线电话，如果仍由总线来转接，总线接线员可能忙不过来，或者由于电话过多而使电话线路总处于占线状态而延误了处理时间。这里还有一个问题，对于那些知名的服务，在用户主机应用进程中仍然非常多，但也并不是所有的应用进程总那么繁

忙，如有的主机可能仅担当少数几种服务器角色，而绝大多数知名服务在这台主机中并没有实际上被提供。因此，此时仍需要对这些知名服务提供集中的管理，以减少服务器资源的浪费。

为了解决上述问题，又推出了一种新的方案——“名称服务器”（name server），有时也称“目录服务器”（directory server），也就是我们通常所说的DNS服务器。它也是“初始连接协议”方案的一种。名称服务器总是在监听一个特定的知名服务的端口（也就是我们通常所说的“常规端口”）。在这种方案中，为了找到与给定的服务名字相对应的TSAP地址（端口），用户需要与名称服务器建立一个连接，然后用户发送一条消息指定它想要的服务名字，名称服务器则发回相应的TSAP地址。随后，用户将它与名称服务器之间的连接释放，再与期望的服务建立一个新的连接。名称服务器为用户进程分配TSAP地址的整体流程与上面介绍的进程服务器为用户进程分配TSAP地址的流程类似，如图10-20所示，只是名称服务器的效率更高而已。

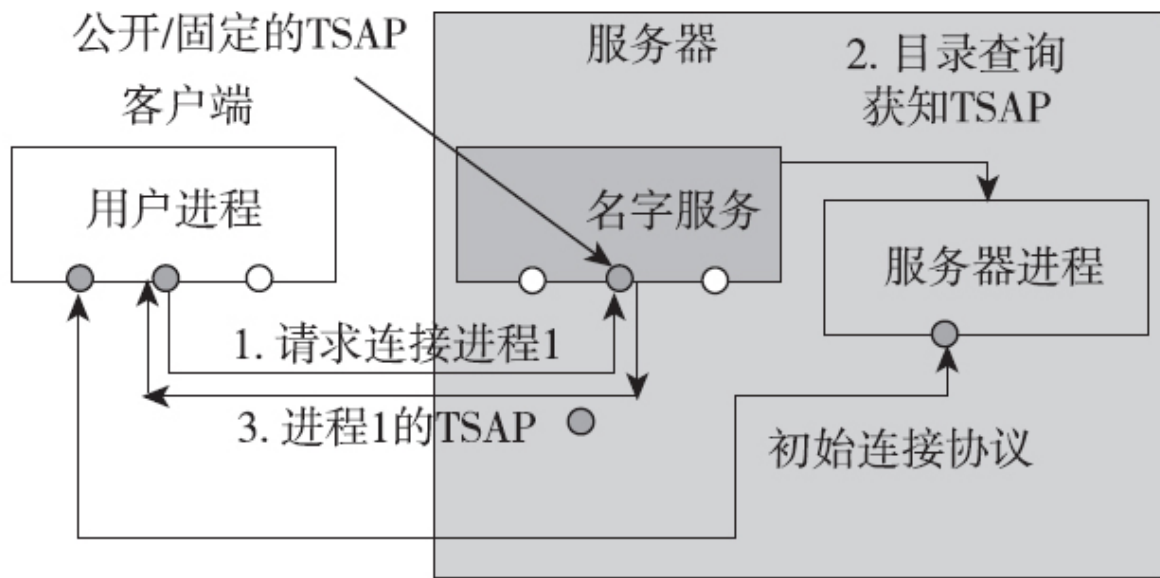


图 10-20 名称服务器为用户进程分配TSAP地址的基本流程

名称服务器之所以知道这些知名服务的TSAP地址，是因为在一个新的服务被创建时必须向名称服务器注册，并把它的服务名字和TSAP地址告诉名称服务器。名称服务器将把这份信息记录到它的内部数据库中，以后当用户查询时就知道了。例如，在配置DNS服务器时，会把这台机器上所配置的各服务器角色的对应服务记录添加到DNS服务器中，这些记录中会记录对应的服务器名称、所用端口信息。

10.2.2 传输连接建立

通过本章前面介绍的内容可知，表面上看起来传输连接的建立过程很简单，用户端只需调用T_CONNECT.request和T_CONNECT.confirm原语，服务器端只需调用T_CONNECT.indication和T_CONNECT.response原语，但其实传输连接的建立非常烦琐，因为它受到网络延迟、子网分组存储、重复分组接收等诸多因素影响。下面先看一下正常情况下基本的传输连接过程。

1.正常传输连接的建立

在正常情况下，传输连接的建立过程比较简单，所用到的传输服务原语是上文介绍的T_CONNECT.request、T_CONNECT.indication、T_CONNECT.response、T_CONNECT.confirm，所用到的TPDU主要有CR TPDU和CC TPDU两个。传输连接建立的基本过程如图10-21所示，描述如下。

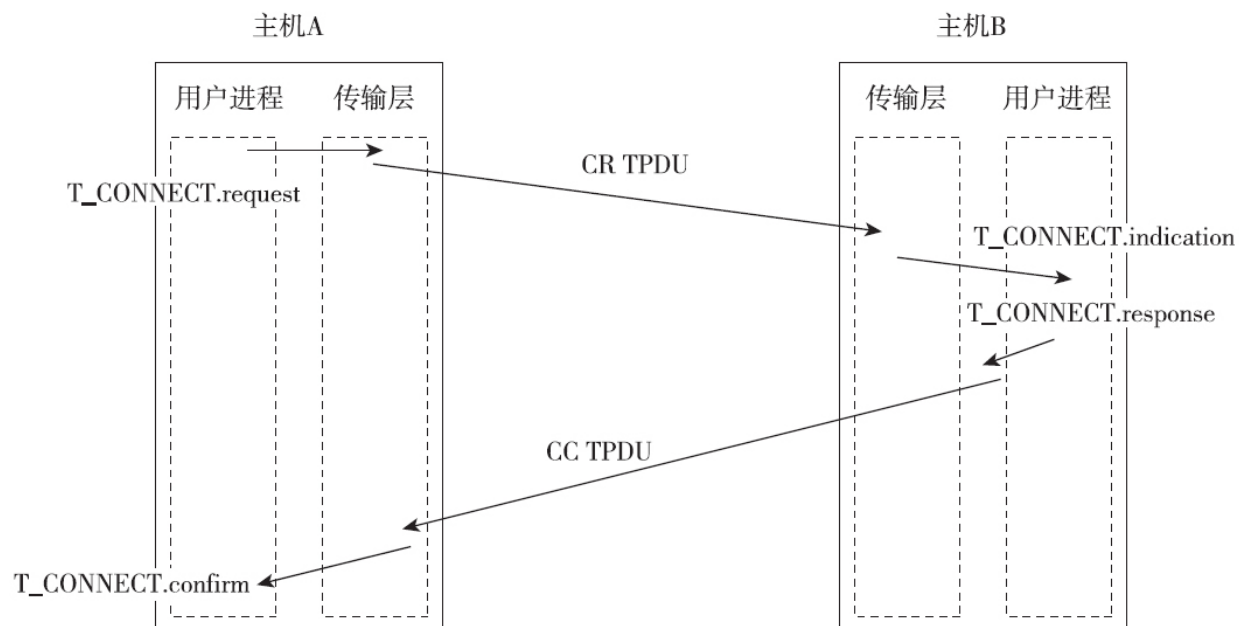


图 10-21 传输连接建立的基本流程

1) 客户端主机A用户实体中的对应用户进程调用传输层的T_CONNECT.request服务原语，由所使用的传输层协议向服务器端主机B发送一条CR（连接请求）TPDU，请求建立一条传输连接。

说明 在OSI/RM体系结构中，用户实体位于“会话层”中，而在TCP/IP体系结构中，用户实体位于“应用层”中，下同。

2) 主机B传输层在收到来自主机A的这条连接请求TPDU后，通过调用T_CONNECT.indication服务原语，告诉自己的用户实体中的对应用户进程：有用户想与自己建立传输连接。

如果传输服务提供者（主机B）提出降低服务质量（如较小的吞吐量、较长的时延、较高的差错率、较低的优先级等）要求，那么这时

一些新的参数会出现在T_CONNECT.indication原语中。

3) 如果接受连接请求, 则主机B用户实体调用传输层的T_CONNECT.response服务原语, 然后通过传输层协议向主机A发送一条CC (连接确认) TPDU。

如果传输服务用户 (主机A) 提出降低服务质量要求, 其参数将出现在T_CONNECT.response原语中。最后, 通过下一步发送的CC TPDU将参数返回到源端的传输用户。

4) 主机A在收到来自主机B的CC TPDU后, 通过调用传输层的T_CONNECT.confirm服务原语, 向自己的用户实体中的源用户进程确认连接已建立, 正式进入数据发送阶段。

从以上的步骤可以看出, 好像传输连接的建立过程很简单, 但实际上, 传输连接的建立远不是这样的, 甚至可以用非常烦琐来形容, 主要体现在因网络拥塞而出现多次连接, 或因网络中存在延迟的重复分组而造成重复连接。

2.传输连接拒绝

首先要注意的是, “传输连接拒绝”不是“传输连接释放”, 它是传输连接请求在建立连接过程中被对端拒绝, 此时传输连接根本还没建立起来, 即在建立过程中就中断了。

在以上“传输连接建立”过程中的第3步，如果连接的目的主机B（服务提供者）的传输层可能因源端的服务质量太差，或者自己的线路很忙而要拒绝主机A（服务请求者）的传输连接请求，这时就可以通过发送RJ TPDU来实现了，具体过程如下。

1) 当来自主机A的CR TPDU到了主机B的传输层后，通过调用自己传输层的T_CONNECT.indication原语向对应的用户进程说明有用户需要与它建立传输连接，同时会通过CR TPDU的分析得出源端的服务质量水平。

2) 如果主机B应用进程认为源端的服务质量水平太差，或者应用层目前“工作”实在太繁忙，不接受此次的传输连接请求，则通知传输层向主机A发送RJ TPDU，表示拒绝此次传输连接。

在“传输连接拒绝”过程中的整个服务原语调用和TPDU发送情况如图10-22所示，与图10-21所示的“传输连接建立”过程相比，唯一的区别就是主机B向主机A发送的不再是CC TPDU，而是RJ（拒绝连接）TPDU。当然，主机B的传输层向自己的应用层提交的T_CONNECT.indication原语中的内容也会有所不同，会表示出不愿意接受连接的意愿，尽管最终决定是否接受连接的是应用层的用户进程，而不是传输层。

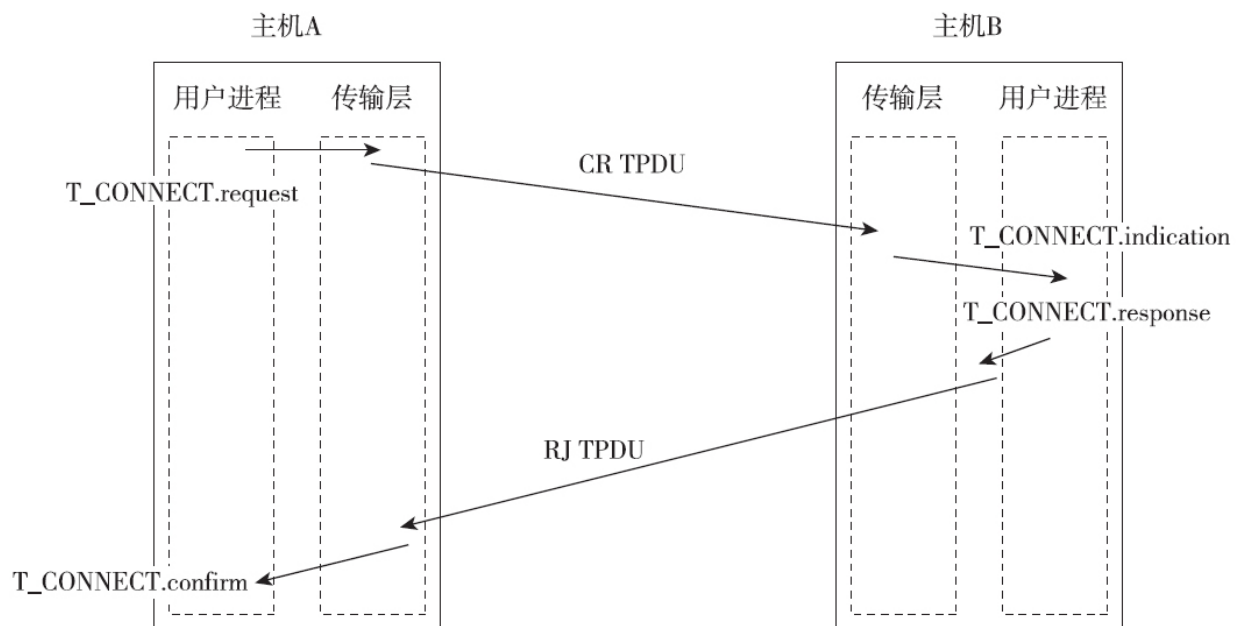


图 10-22 传输连接请求拒绝的基本流程

10.2.3 重复传输连接的解决方法

在传输连接建立过程中，除了大多数可以正常一次连接成功外，还可能因各种原因出现一些非正常连接现象，如因网络拥塞而造成的多次连接，或因子网分组存储而造成的重复连接等。

如果网络非常拥塞，就会导致传输连接建立过程中所传输的确认TPDU分组都无法及时回到发送端，这样一来，传输连接发起方可能需要进行多次连接尝试。网络延迟还可能因为“存储-转发”机制引起，特别是在传输通信两端途经的网络比较多时，由于各子网的分组存储的问题，导致一些TPDU分组没有及时到达，最终造成传输连接发起方会多次重试。但问题远不是重试这么简单，因为这些延迟的分组可能在过一段时间后到达了，于是又会请求建立新的连接，这就出现了重复连接的问题。

对于重复的传输连接问题，目前还没有一种特别有效的解决方法，通常是采取非重复TSAP、超时连接表、分组的TTL机制和三次握手机制等方法尽量避免。

(1) 非重复TSAP

非重复TSAP方法是系统为每次连接赋予一个新的TSAP地址，当连接被释放时将此TSAP废弃。这样一来，就不会出现重复连接的现象。但这种解决方法不适用于那些知名服务的连接，因为像WWW、FTP、POP之类的传输连接是固定使用那些知名的端口进行的，不可能每次都变。

（2）过时连接表

过时连接表是为每个连接分配连接标识符（也就是传输连接序列号）的，每新建一个传输连接，序列号增加1，并且存入到每个TPDU（也包括请求建立连接的TPDU）中。当每个连接被释放时，将此连接信息存入过时连接表，然后在每个新的连接请求到达时，根据标识符核对过时连接表，如果是已过时的序列号，则视为重复连接。但这种方法需要每个传输实体无限期地维护一张历史记录表，不适应突变情况，如机器重启、系统崩溃等。

（3）分组的TTL机制

分组的TTL机制是先定义和计数每个分组的生存时间TTL，如果因超时未达目的地，则视为陈旧分组，将被抛弃。此机制同样也存在一些问题，如消除重复连接请求需要依赖子网完成、网络层分组传输本身的不可靠性等。

（4）三次握手机制

所谓“三次握手”是指在建立连接时，发送端发送CR TPDU请求建立一个连接（第一次握手）；接收端收到CR TPDU后发送一个CC TPDU进行应答（第二次握手），在应答分组中宣告了自己的初始连接序列号；发送端收到应答分组后发送第一个DT TPDU，对接收端初始序列号进行确认（第三次握手）。这样，一个连接才能建立起来，并且双方可以使用不同的起始序号。整个三次握手过程如图10-23所示，简单描述如下（在TCP/IP体系结构中的TCP传输连接的建立也采用“三次握手机制”，具体内容将在本章后面介绍）。

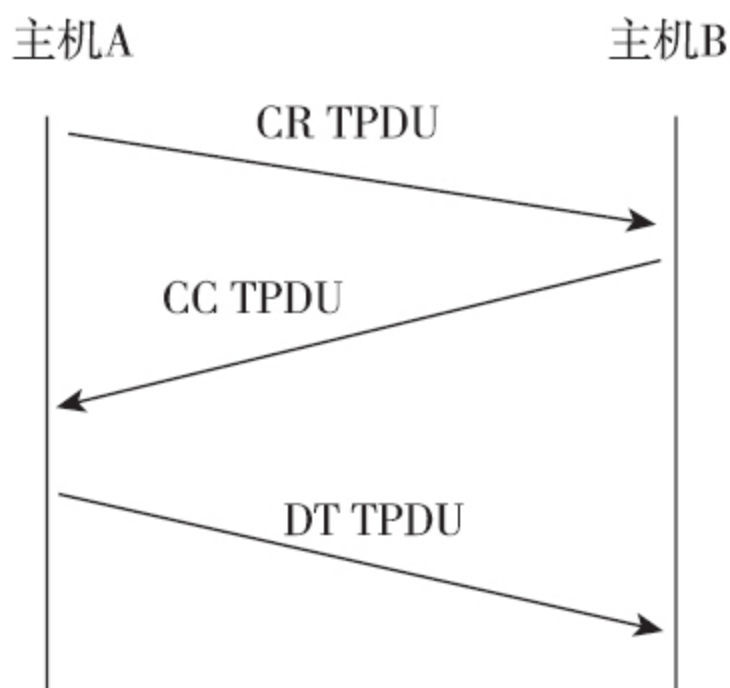


图 10-23 正常情况下的传输连接三次握手过程

在正常情况下，主机A向主机B发送一个“源参考”为i（“目的参考”为0）的CR TPDU请求以建立连接；主机B收到CR TPDU后发送一

个“源参考”为j（“目的参考”为i）CC TPDU应答分组，对主机A发来的连接请求进行确认；主机A在收到的CC TPDU“源参考”字段后就获知主机B的传输连接参考，把它作为“目的参考”字段值发送第一个携带少量数据的DT TPDU（其中包括了DT TPDU的序列号），同时对主机B选择的初始连接序号进行确认。

其实三次握手连接的原理很简单，就像我们平时聊天一样，首先是由发起人说“Hello”（打招呼通常预示着要向对方说话），当对方听到这个招呼后，如果确认此时可以与发起方聊天，则也会发出一个招呼“Hello”。这里有两重意义，首先是证明他已听到发起方的招呼，另一方面也说明目前自己可以与发起方聊天。最后，发起方在收到对方的回应招呼后，可以再次选择，如果恰在此时又有其他事不能与对方聊天了，当然可以放弃，如果认为条件成熟，则可以说“OK, Begin”（好的，开始吧），接下来就是正式的聊天了。

在出现延迟的重复CR TPDU的情况下，在网络中间延迟发送的一个源参考为i的重复CR TPDU到达主机B后，请求在已释放的连接上建立连接，主机B发送一个CC TPDU应答分组对原来源参考为i的请求进行应答，同时在“源参考”字段中宣告自己的传输连接号为j；主机A通过检查可知主机B的应答分组中确认的源参考为i的连接原来已经确认，便认为该应答是过时的，并向主机B发送一个RJ TPDU拒绝连接，整个过程如图10-24所示。这样，通过三次握手机制可以有效地减

少因延迟的重复CR TPDU分组而带来的负面影响，从而提高建立连接的安全性和可靠性。

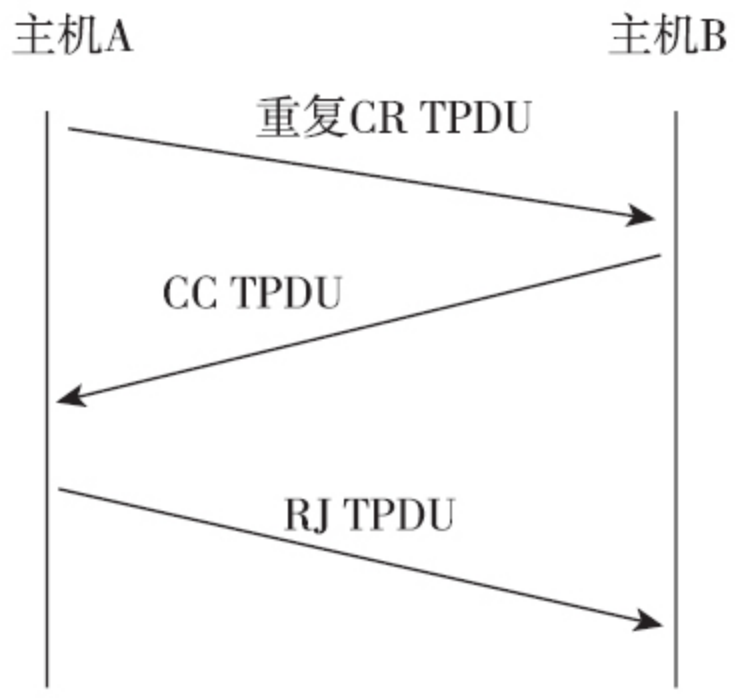


图 10-24 重复连接的三次握手过程

10.2.4 数据传输

一旦传输连接建立，传输协议即进入了正式的数据传输阶段。ISO 规定了正常数据传输和加速数据传输两种类型的服务，其中分别涉及了T_DATA.request、T_DATA.indication、T_EXPEDITED_DATA.request和T_EXPEDITED_DATA.indication四个传输服务原语，以及DT TPDU、AK TPDU、ED TPDU、EA TPDU四个TPDU。图10-25表示了4种数据传输时的服务原语和TPDU的使用情况，具体过程描述如下。

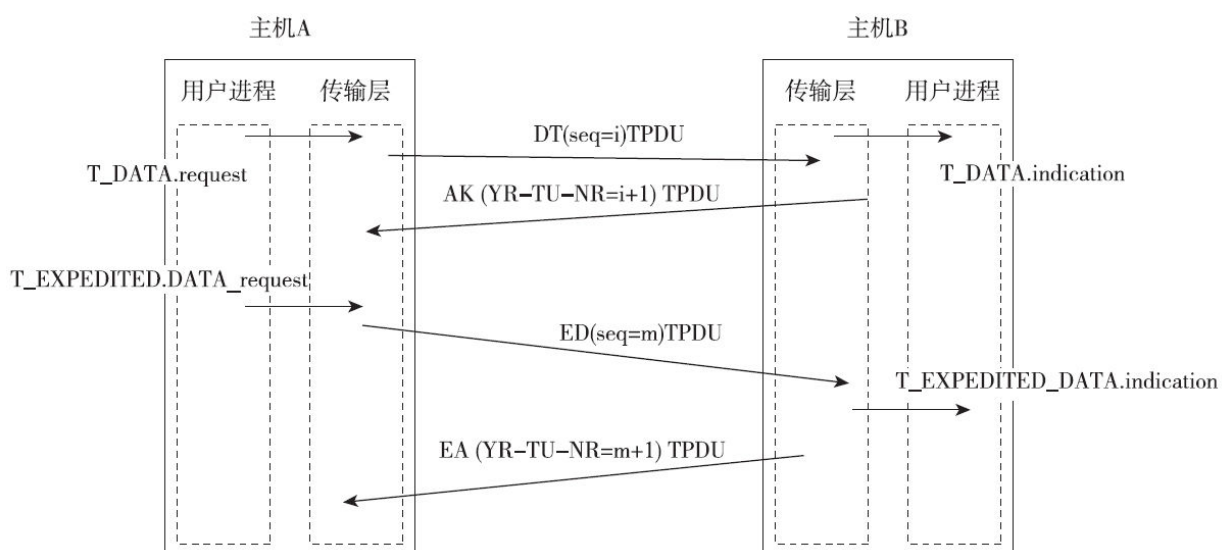


图 10-25 数据传输基本流程

1) 当传输连接建立好并需要发送普通数据时，主机A的用户实体调用T_DATA.request传输服务原语（当发送加速数据时，调用T_EXPEDITED_DATA.request传输服务原语），指示传输层协议开始进

行数据发送。普通数据是以DT TPDU进行发送的（其中“TPDU序列号”字段值假设为 i ）；加速数据是以ED TPDU进行发送的（其中“TPDU序列号”字段值假设为 m ）。

说明 如果发送的数据是经过分段的，每个数据段的TPDU都有子序列号（在“可变部分”中），则将最后一个数据段中的EOT（传输结束）置1；如果不是最后一个数据段，则该位置0。

2) 当数据到达主机B的传输层时，普通数据通过T_DATA.indication服务原语（加速数据通过T_EXPEDITED_DATA.indication服务原语）向用户实体指示有数据到达。

3) 如果收到的是普通数据，主机B发送AK TPDU进行确认，其中“你的TPDU序列号”（YR-TU-NR）字段为希望收到的下一个数据或加速数据TPDU的序列号，等于 $i+1$ ，但没有“TPDU序列号”字段值。如果收到的是加速数据，则主机B发送EA TPDU进行确认，其中“你的TPDU序列号”（YR-TU-NR）字段为希望收到的下一个数据或加速数据TPDU的序列号，等于 $m+1$ 。

10.2.5 传输连接释放

传输连接的释放分为“对称释放”和“非对称释放”两种。“对称释放”是在各自独立发出释放连接请求，收到对方的释放确认之后才可释放连接；“非对称释放”是在发送释放请求后单方终止连接，但这样做有可能丢失对方发送的数据。

OSI/RM体系结构的传输连接释放也采用三次握手协商方式（TCP/IP体系结构中的TCP传输连接释放是采用“四次握手机制”的，具体内容将在本章后面介绍），其中也分“正常释放连接”和“非正常释放连接”两种情况。“非正常释放连接”又包括以下3种情况：最后的确认TPDU丢失、应答TPDU丢失、应答及后续释放请求DR丢失。本节以对称释放方式为例进行介绍。

正常情况下的三次握手连接释放过程如图10-26所示。注意，图10-26中省略了释放连接请求方用户实体调用的T_DISCONNECT.request传输服务原语，以及释放连接接受方用户实体调用的T_DISCONNECT.indication服务原语。其中使用的TPDU有两类：用于请求释放连接的DR TPDU、用于确认释放连接的DC TPDU。

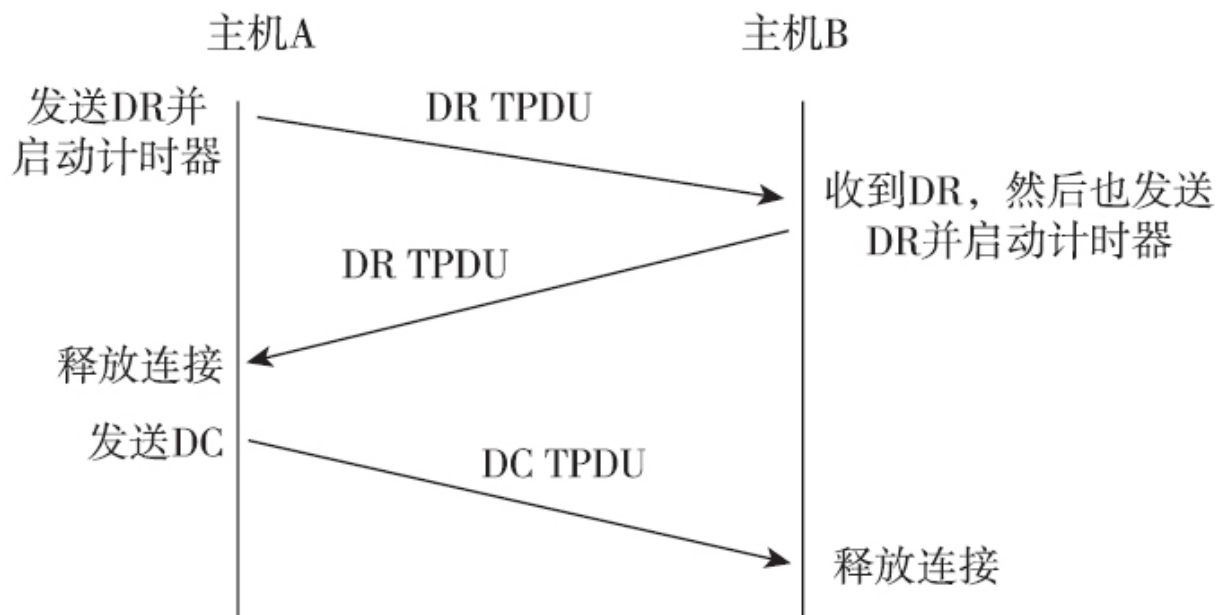


图 10-26 正常情况下的三次握手连接释放过程

如果最后的DC TPDU丢失，则三次所握手协商过程如图10-27所示，最终的释放连接请求会因超时而自动释放。

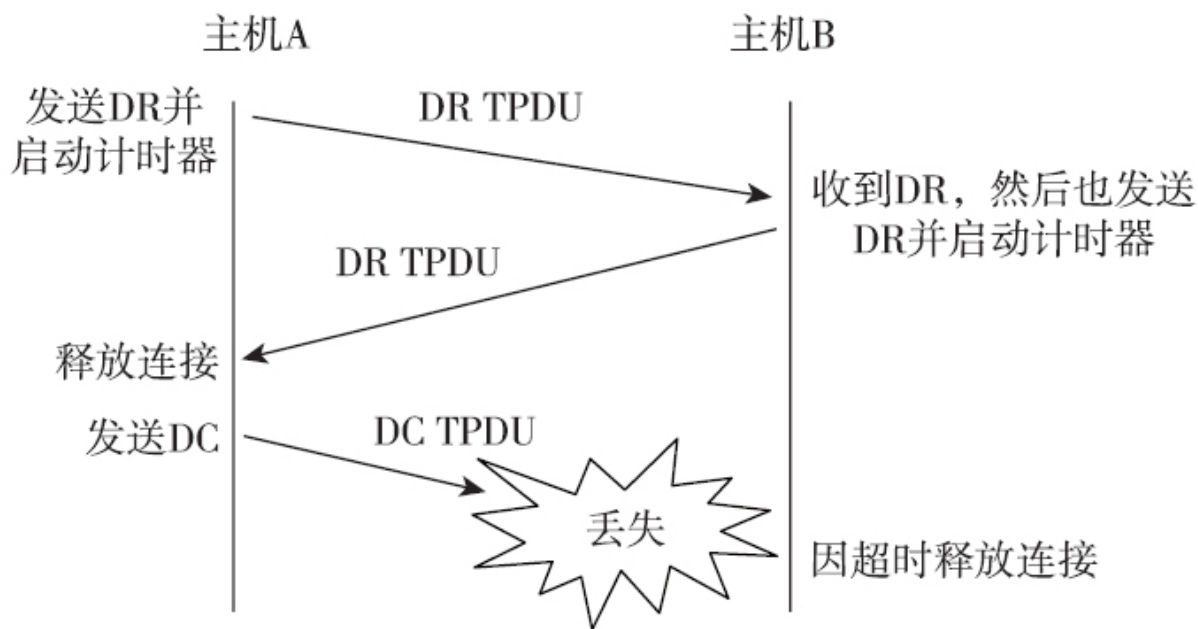


图 10-27 “最后的DC TPDU丢失”情况下的非正常连接释放过程

如果另一方的DR TPDU丢失，则发起释放连接请求的一方会再次重新发送DR TPDU，请求释放连接。连接释放过程如图10-28所示，在请求方重发DR TPDU后释放连接。

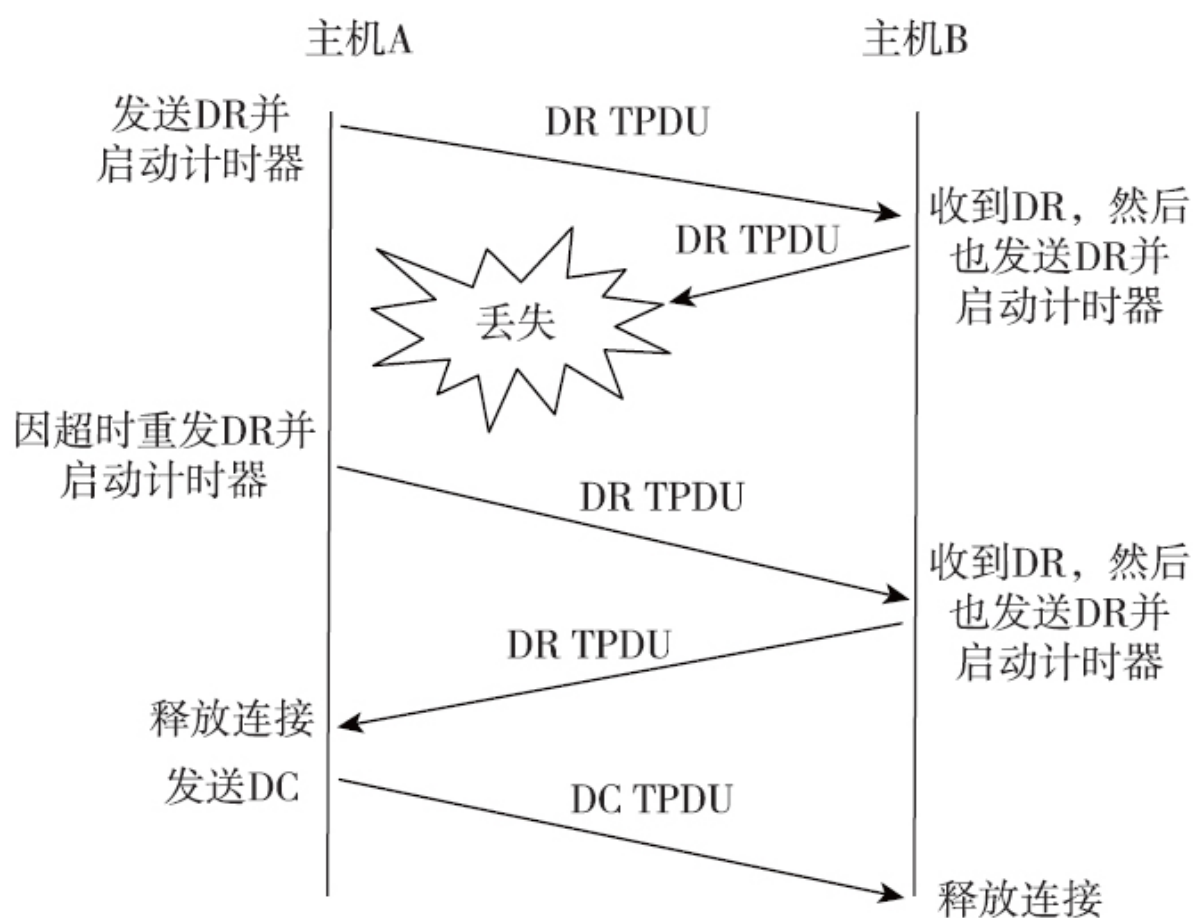


图 10-28 另一方DR TPDU丢失情况下的非正常连接释放过程

如果另一方的DR TPDU和请求方重发的DR TPDU均发生丢失，而且重发多次仍不能成功，则最后因传输超时而导致连接自动释放，过程如图10-29所示。

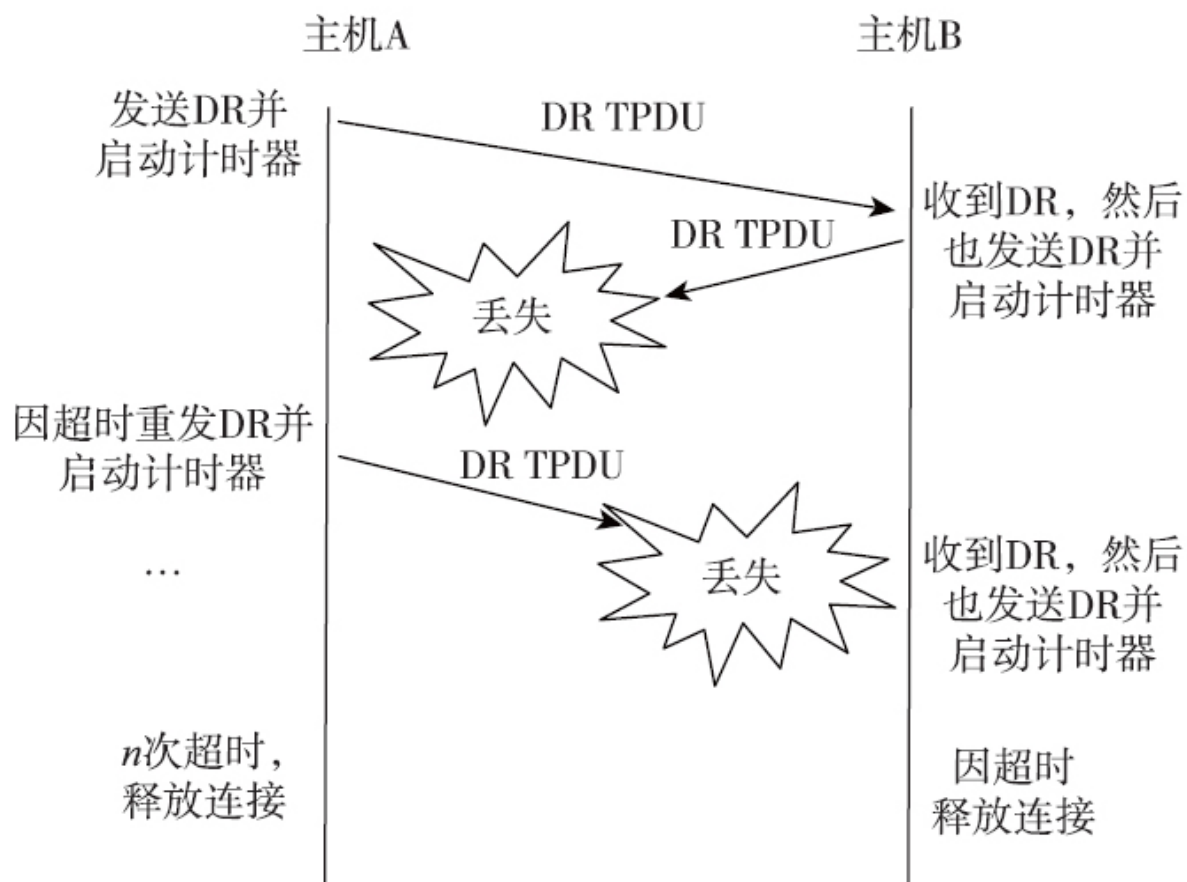


图 10-29 另一方DR TPDU和重发DR TPDU均丢失情况下的非正常连接释放过程

10.2.6 流量控制

传输层协议采用的是在本章前面已多次提到的信用量（CDT）分配的窗口机制，不同于数据链路层协议流量控制方案中的窗口尺寸固定的滑动窗口机制，因为传输层中的CDT值是随着数据的传输过程而不断变化的（TCP/IP体系结构中的TCP的流量控制原理也是这样的，本章后面将具体介绍）。在传输层的CR、CC、AK三类TPDU头部中，均有一个CDT字段（参见10.1.5节和10.1.6节，相当于本章后面介绍的TCP数据段中的“窗口大小”字段），接收端利用该字段向发送端通报当前还可接收的TPDU数量，即接收缓冲区大小，发送端则按当前接收缓冲区容量发送适量的TPDU。

在建立连接时，双方通过CR TPDU和CC TPDU中的CDT字段捎带着相互通告各自的初始窗口大小，即初始信用量。在数据传输过程中，发送端按接收端发回的AK TPDU中的CDT值发送一定的数据量；接收端可根据接收缓冲区的使用状况动态地调整接收窗口大小，并在发送AK TPDU进行确认时捎带着将新的窗口大小通告给发送端。发送端将按新的接收窗口尺寸来调整发送窗口大小，接收端也用新的接收窗口大小来验证新到达数据分组的可接受性。下面举两个例子来说明一下，为了便于说明，假设双方的初始窗口大小相同。

1.窗口大小为1的数据传输过程

在传输连接建立后，当发送端发送的CR TPDU和接收端发送的CC TPDU中的CDT值均为1（0001）时，表明发送端和接收端的窗口大小为1，即发送端发送一个TPDU后必须等待在接受了该TPDU的应答TPDU后才能继续传送新的TPDU；接收端也类似，即仅当发送端发送一个TPDU后，接收端才能向发送端返回一个确认TPDU。也就是说，发送端发送一个，接收端必须确认一个，如图10-30所示，其实这与数据链路层的“停-等”窗口协议原理是一样的。

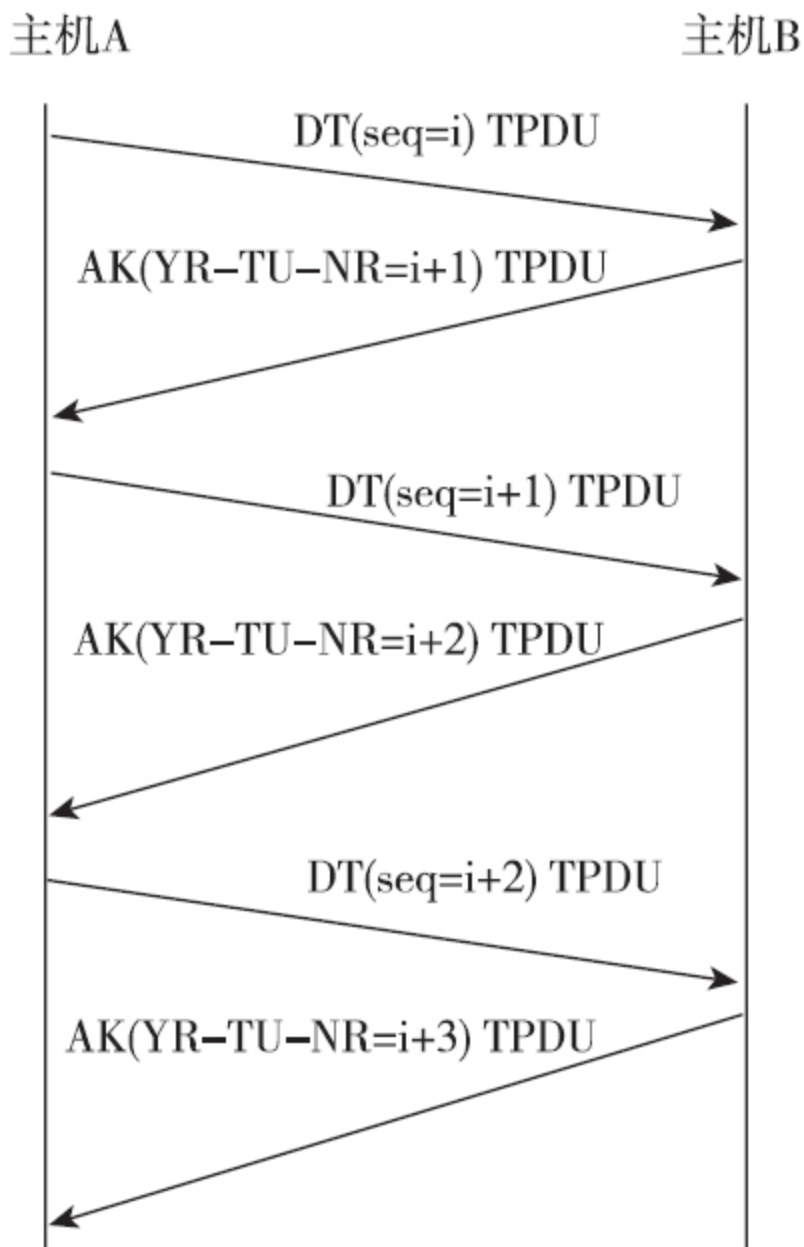


图 10-30 窗口大小为1时的数据传输过程

假设发送端主机A和主机B的初始TPDU序列号分别为i和j，具体的步骤描述如下：

1) 主机A向主机B发送第一个DT TPDU, 序列号为 i ($\text{seq}=i$)。因为窗口大小为1, 所以必须等待对方收到并发回确认后才能继续发送下一个DT TPDU。

注意 仅DT TPDU和ED TPDU中有“TPDU序列号”字段, 其他TPDU均没有。

2) 主机B在收到序列号为 i 的DT TPDU并向用户实体提交(清空缓存)后向主机A发送第一个确认AK TPDU, 其中“你的TPDU序列号”字段值为 $i+1$, 即 $\text{YR-TU-NR}=i+1$ (AK TPDU和EA TPDU等均无“TPDU序列号”字段)。

3) 因为主机B已把收到的DT TPDU向用户实体提交了, 所以缓存为空, 可继续向主机B发送一个DT TPDU, 序列号为 $i+1$ ($\text{seq}=i+1$)。此时同样因窗口大小为1, 不能继续发送, 需等待主机B的确认TPDU。

4) 主机B在收到序列号为 $i+1$ 的DT TPDU并向用户实体提交(清空缓存)后向主机A发送第一个确认AK TPDU, 其中“你的TPDU序列号”字段值为 $i+2$, 即 $\text{YR-TU-NR}=i+2$ 。

以后的步骤就与前面介绍的一样了, 主机A发送一个DT TPDU后就停下来, 只有在收到主机B的确认TPDU后才继续发送下一个DT TPDU。在这种情况下特点是, 只有发送端收到一个确认TPDU, 才

可以再发送一个新的DT TPDU。当然，这种情况比较少见，因为这种窗口机制不能有效地利用信道资源。

2.窗口大小为3的数据传输过程

适当调整窗口大小可以不仅使接收端和发送端速率相匹配，还可以充分利用现有带宽资源，提高传输效率。如果在CR TPDU和CC TPDU中设置CDT为3，即双方的窗口大小均为3，则发送端可以在第一次连续发送3个DT TPDU后再停下来等待接收端的一个或多个确认信息，接收端也类似。但是发送端和接收端并不是每次都可以连续发送3个TPDU，要视对方返回的确认TPDU确认数多少而定，其实也就是要视对方当前可用缓存空间大小而定。

图10-31是在窗口大小为3的情况下的数据传输过程，具体描述如下：

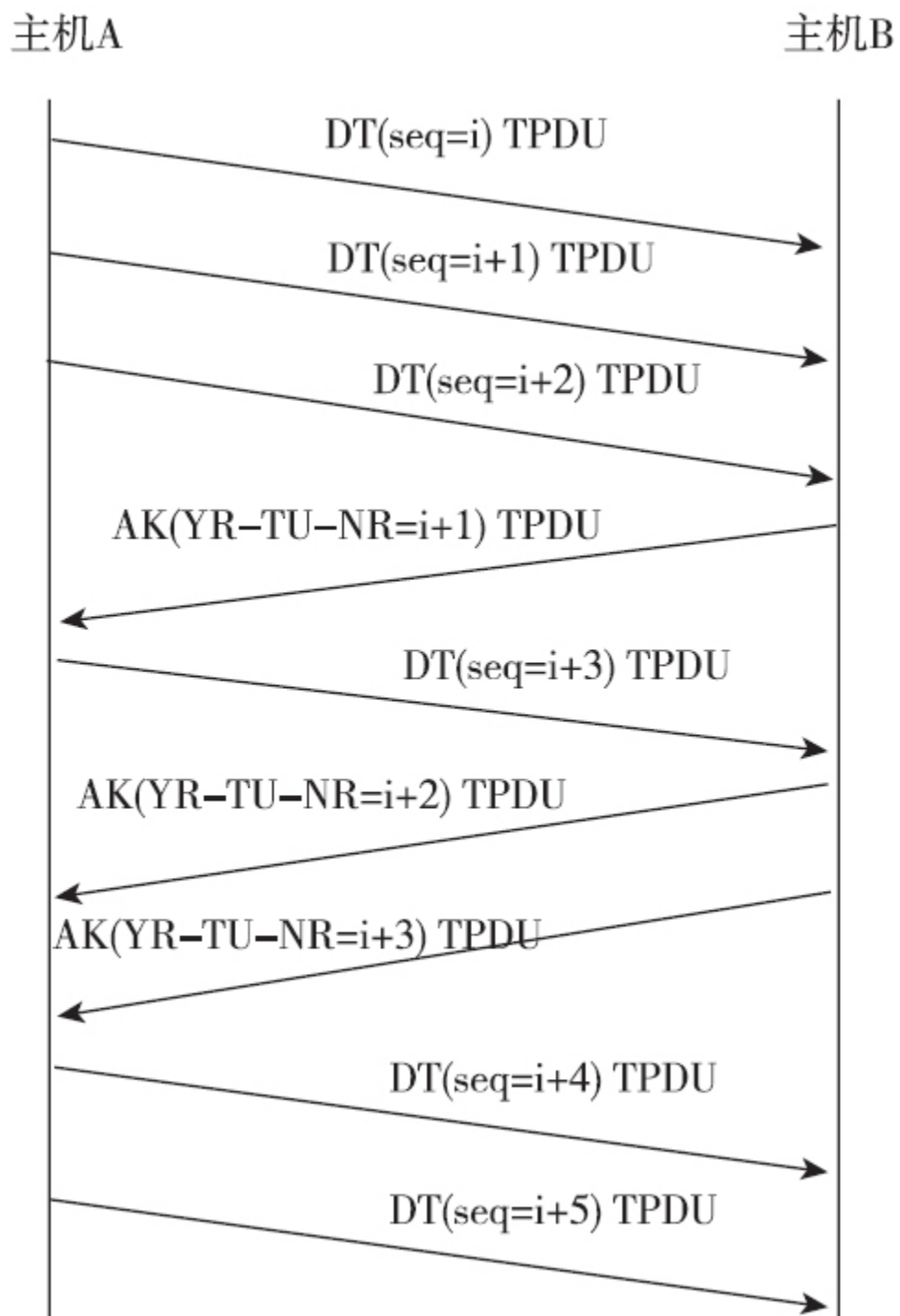


图 10-31 窗口大小为3时的数据传输过程

1) 主机A向主机B连续发送3个DT TPDU，序列号分别为i (seq=i)、i+1 (seq=i+1) 和i+2 (seq=i+2)。因为窗口大小为3，所以必须等待对方收到并发回确认（相当于清空部分或全部缓存）后才能继续发送下一个DT TPDU。

2) 主机B在收到前面3个DT TPDU后，依次向用户实体提交，并依次向主机A发送确认AK TPDU。现假设主机A收到了主机B的第一个AK TPDU，“你的TPDU序列号”字段值为i+1。此时，在AK TPDU中CDT为1，表示可以再接收一个TPDU。

3) 在收到主机B发来的第一个AK TPDU后，看到其中的CDT值为1，于是马上向主机B发送下一个DT TPDU，序列号为i+3 (seq=i+3)。此时，还没有收到新的来自主机B的AK TPDU，表明对方没有可用缓存空间了，需要停下来，继续等待来自主机B新的确认TPDU。

4) 现假设主机A在很短的间隔时间内几乎同时收到了两个AK TPDU，“你的TPDU序列号”字段值分别为i+2和i+3。主机A从后面收到的AK TPDU中的CDT值得知主机B目前可以再接收两个TPDU，于是又连续向主机B发送两个DT TPDU，序列号分别为i+4 (seq=i+4)、i+5 (seq=i+5)。

随后的步骤与前面介绍的类似。总之，需要注意的一个原则是，发送端的主机A收到了来自接收端的主机B的多少个确认TPDU，就可继续发送多个DTP TPDU，发送窗口大小不是固定的。

在传输层中，还具有数据缓冲的功能，虽然这也与数据链路层的数据缓冲功能类似，但所采取的措施不一样。传输层的缓存策略和管理方法包括缓存方式和策略、发送端申请/接收端分配、接收端在应答中夹带新的分配信息、阻塞和潜在死锁等几个方面，在此不进行介绍了。

10.2.7 多路复用

在传输层也可以采取多路复用（**session multiplexing**）技术来提高传输连接的利用率。传输层的多路复用可以由多个传输连接复用同一个网络连接的向上多路复用方式，也可以是由一个传输连接在多个网络连接上循环传输的向下多路复用方式。

假设一台机器只有一个网络地址可以使用，那么这台机器上的所有传输连接都必须使用这一个网络地址，这就是向上多路复用方式，如图10-32所示。当一个TPDU进来的时候，传输实体需要用一种方法来指明应该将它交给哪个应用进程，这个方法就是利用不同传输连接所而使用不同的端口号。

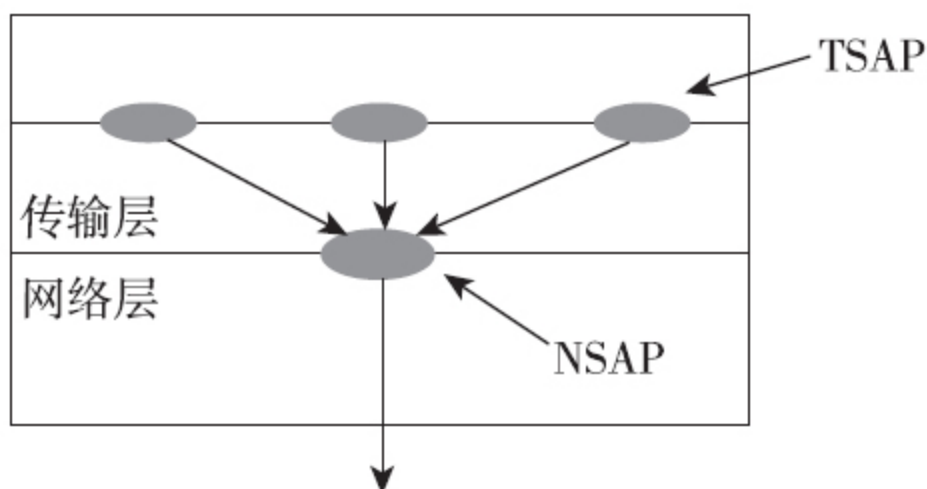


图 10-32 向上多路复用示意图

还有一种情况，在一个子网中使用了虚电路，并且每条虚电路上有最大的数据传输速率限制，当某个传输连接所需的带宽超过某一条虚电路所能提供的带宽时，就需要将这个传输连接以轮循的方式把数据流分配在不同的网络连接上，这就是向下多路复用方式，如图10-33所示。向下多路复用方式的典型应用是ISDN用户线路，因为普通的家庭ISDN中每条线路只有64kbit/s，通过复用两条线路，最终达到128kbit/s的带宽。

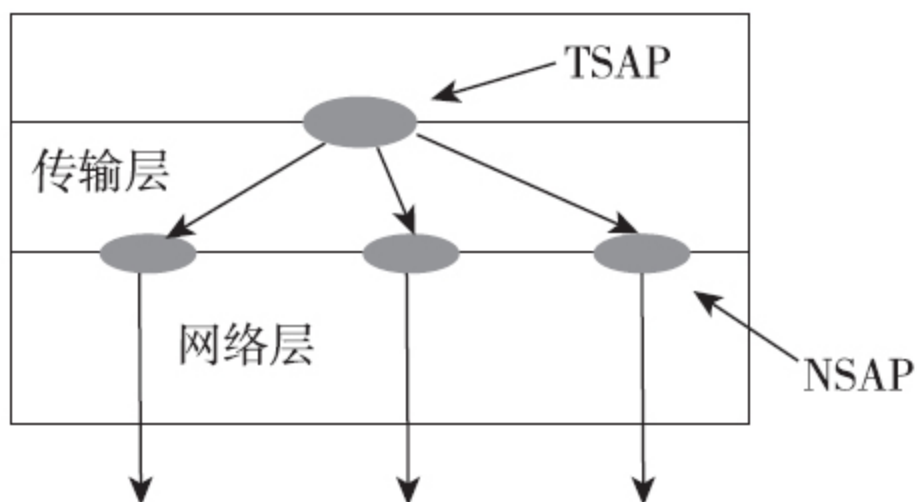


图 10-33 向下多路复用示意图

10.2.8 崩溃恢复

在传输连接中由于遇到主机或路由器不能正常工作而造成TPDU传输过程中断的现象称为崩溃。崩溃恢复就是要恢复到崩溃前的工作状态，继续TPDU传输。要实现这样一项功能，必须解决在崩溃前发送的最后一个TPDU是否需要重传的问题。

如果网络层提供数据报服务，就需要传输实体留有丢失的TPDU的副本；如果网络层提供面向连接的服务，则重建新的虚电路，重发丢失的TPDU。要从主机或服务器崩溃中恢复，存在着许多复杂的问题，在此不进行讨论。

总体来说，从第N层崩溃中恢复只能由第N+1层来完成，并且只能第N+1层留有足够状态信息的情况下才能完成。

对于传输层崩溃恢复，需要先了解在崩溃前发送主机和接收主机的状态，以及发送主机对最后TPDU的处理。崩溃恢复工作必须由更高层次（应用层）来完成，具体的连接恢复机制参见10.4节中介绍的TCP连接可靠性内容。

10.3 TCP概述

前面主要介绍了OSI/RM体系结构中的传输层服务，本节要具体介绍TCP/IP体系结构中的主要传输层协议——TCP（传输控制协议），其中包括TCP的一些主要特性和在本章前面介绍的各种传输服务功能实现的原理。

10.3.1 TCP的主要特性

TCP是TCP/IP体系结构中最主要的传输层协议，在功能上与本章前面介绍的OSI网络中的TP4类传输层协议类似，但TCP只能提供面向连接的传输服务，而且TCP还具有以下主要特性：

（1）面向连接的传输协议

也就是说，应用程序在使用TCP之前，必须先建立TCP传输连接，在传输数据完毕后，必须释放已建立的TCP传输连接。

（2）仅支持单播传输

每条TCP传输连接只能有两个端点，只能进行点对点的数据传输，不支持多播（multicast）和广播（broadcast）传输方式。

说明 这里所说的TCP传输连接的“端点”不是主机、主机的IP地址、应用进程、传输层协议端口，而是套接字（socket）。套接字是IP地址和端口号的组合，中间用冒号或逗号分隔，如

（192,168.10,80）。每个TCP传输连接有两个端点，也就是有两个套接字，即“源IP地址和端口号”组合和“目的IP地址和端口号”组合，可表示为 { socket1,socket2 }，或者 { (IP1,Port1), (IP2,Port2) }。

（3）提供可靠的交付服务

也就是说，通过TCP连接传送的数据可以无差错、不丢失、不重复，且按时序到达对端。

（4）传输单位为数据段

TCP仍采用了传统的“数据段”作为数据传输单元。由于数据段大小受应用层传送的报文大小和所途经网络中MTU值大小决定，所以每次发送的TCP数据段大小是不固定的。在一个具体的网络中，有一个MSS（Maximum Segment Size，最大数据段大小），最小的数据段可能仅有21字节（其中20字节属TCP数据段头部，数据部分仅1字节）。

（5）仅一种TPDU格式

TCP的数据传输数据单元称为“数据段”，其实也可以按照OSI/RM中的说法称为TPDU。在TCP中只有一种TPDU，而不会像OSI/RM中有

很多种不同格式的TPDU（具体参见10.1.5节~10.1.7节），因为在TCP数据段头部已包括了各种TPDU所需的特征字段，主要是通过其中的多个控制位来实现的，具体将在说明TCP数据段格式时介绍。

（6）支持全双工传输

TCP允许通信双方的应用程序在任何时候都能发送数据，因为TCP连接的两端都设有发送和缓存，用来临时存放双向通信的数据。当然，TCP可以立即发送一个数据段，也可以缓存一段时间以便一次发送更多的数据段（最大的数据段大小取决于MSS）。

（7）TCP连接是基于字节流的，而非报文流

TCP不像UDP那样以一个个报文独立地进行传输，而是在不保留报文边界的情况下以字节流方式进行传输。

（8）每次发送的TCP数据段大小和数据段数都是可变的

在TCP中，因为每次发送多少字节的数据不是固定的，也不是由主机当前可用缓存决定的，而是根据对方给出的窗口大小和当前网络的拥塞程度来决定的（本章后面将要介绍的UDP发送的报文长度是由应用进程给出的），所以每次可以发送的TCP数据大小是不固定的。

TCP的传输单元是数据段，数据段又是在应用层传送的数据基础上进行封装的，每次的TCP数据段大小取决于不同的应用进程数据。

当然，如果应用层传送的报文太长，超出了MSS，则需要对报文进行分段，总的来说，每次传输的TCP数据段大小是由应用层数据和MSS双重决定的。

这样一来，每次可以发送的TCP数据段数也是不固定的，可以一次仅发送一个TCP数据段，也可以一次发送多个TCP数据段，只要在当前可用的“发送窗口大小”限制之内即可。另外，如果应用进程传送到TCP缓存中的数据太长，TCP可以对它进行分段，反之，如果传到TCP缓存中的数据太小，则TCP会等待缓存中有足够多的数据后再组装成一个数据段一起发送。

10.3.2 TCP数据段格式

前面讲过，TCP的协议数据单元仍采用传统意义上的叫法，称为Segment（分段），而没有像OSI/RM中的那样称为TPDU，当然两者本质上其实是一致的。所使用的TCP软件决定了数据段的大小，可以将多次写操作中的数据累积起来放到一个数据段中，也可以将一次写操作的数据分割成多个数据段。有两个因素决定了数据段的大小：一是每个TCP数据段的大小必须符合IP数据包的65515字节的有效载荷大小限制要求；二是每个网络都有一个MTU值，因此，每个TCP数据段必须符合MTU的限制要求。

TCP通过数据段的交互来建立连接、传输数据、发出确认、进行差错控制、流量控制及关闭连接。整个TCP数据段也分为“数据段头”和“数据”两部分（这也是绝大多数报文封装的方式），所谓“数据段头”就是TCP为了实现端到端可靠传输而加上的TCP控制信息，而“数据”部分则是指由高层（即TCP/IP体系结构中的“应用层”）来的用户数据。但由于TCP中只有一种TPDU格式，所有类型的数据段格式都统一在如图10-34所示的TCP数据段格式中，不同类型数据段是通过其中的多个控制位来实现的。下面是其中各字段的具体说明。

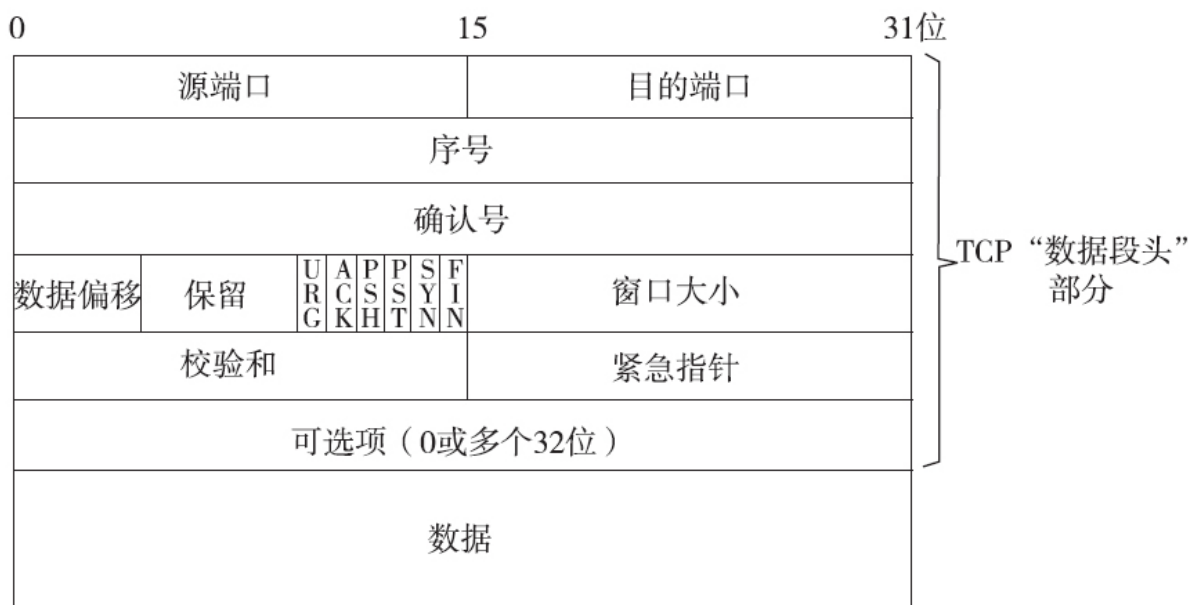


图 10-34 TCP数据段格式

（1）源端口和目的端口

源端口和目的端口分别代表呼叫方和被叫方的TCP端口号，各占16位。一个端口与其主机的IP地址就可以完整地标识一个端点了，也就是构成了前面所说的套接字（Socket）。

（2）序号（Sequence Number）

序号指TCP数据段中的“数据”部分（不包含“数据段头”部分）的第一个字节的编号，占32位。在一个TCP连接中，传送的数据字节流中的每一个数据字节都要按顺序进行编号，在“数据段头”中标识的只是每个数据段的第一个数据字节的编号。整个要传送的字节流的起始序号必须在连接建立时设置。例如，一个数据段的“序号”字段值是101，而

该数字段中共有100字节，表明本数据段的最后一个字节的编号是200。这样一来，下一个数据段的“序号”字段值应该是201，而不是102，这点要特别注意。

(3) 确认号 (Acknowledgment Number)

确认号指期望接收到对方下一个数据段中“数据”部分的第一个字节序号，占32位。注意，“确认号”不是代表已经正确接收到的最后一个字节的序号。例如，主机B已收到主机A发来的一个数据段，其序号值是101，而该数据段的长度是100字节。这表明主机B已收到主机A前200个字节，下一个期望要收到的数据段的第一个字节的序号应该是201，于是主机B在给主机A发送确认数据段时要把“确认号”设置为201。

“序号”和“确认号”两个字段共同用于TCP服务中的差错控制，确保TCP数据传输的可靠性，具体内容将在本章后面介绍。

(4) 数据偏移

数据偏移指数据段中的“数据”部分起始处距离TCP数据段起始处的字节偏移量，占4位。其实这里的“数据偏移”也是在确定TCP数据段头部分的长度，因为“数据”部分是紧接着数据段头的。因为TCP数据段头中有不确定的“可选项”字段，所以数据偏移字段是非常必要。但要注意的是，数据偏移量是以32位（即4字节）为单位来计算的，而不是以

单个字节来计算的。因为4个比特位可以表示的最大数为15，所以数据偏移量最大为60字节，这也是TCP数据段头部分的最大长度。

(5) 保留 (Reserved)

这是为将来应用而保留的6个比特位，目前应全设置为0。

(6) URG

Urgent Pointer（紧急指针）控制位，指出当前数据段中是否有紧急数据，占1位，置1时表示有紧急数据。紧急数据会优先安排传送，而不会按照原来的排队顺序进行发送，相当于本章前面介绍OSI/RM传输层TPDU时所说的“加速数据”。仅当本字段的置1，后面的“紧急指针”字段才有意义。

(7) ACK

Acknowledgement（确认）控制位，指示TCP数据段中的“确认号”字段是否有效，占1位。仅当ACK位置1时才表示“确认号”字段有效，否则表示“确认号”字段无效，应用层实体在读取数据时可以不关“确认号”字段。

(8) PSH

Push（推）控制位，指示是否需要立即把收到的该数据段提交给应用进程，占1位。当**PSH**位置1时要求接收端尽快把该数据段提交给应用进程，而置0时没这个要求，可以先缓存起来。

（9）RST

Reset（重置）控制位，用于重置、释放一个已经混乱的传输连接，然后重建新的传输连接，占1位。当**RST**位置1时，释放当前传输连接，然后可以重新建立新的传输连接。

（10）SYN

Synchronization（同步）控制位，用来在传输连接建立时同步传输连接序号，占1位。当**SYN**位置1时，表示这是一个连接请求或连接确认报文。当**SYN=1**，而**ACK=0**时，表明这是一个连接请求数据段（相当于本章前面介绍的OSI/RM TPDU中的CR TPDU）。如果对方同意建立连接，则对方会返回一个**SYN=1**、**ACK=1**的确认（相当于本章前面介绍的OSI/RM TPDU中的CC TPDU）。

（11）FIN

Final（最后）控制位，用于释放一个传输连接，占1位。当**FIN**位置1时，表示数据已全部传输完成，发送端没有数据要传输了，要求释

放当前连接，但是接收端仍然可以继续接收还没有接收完的数据。在正常传输时，该位置0。

（12）窗口大小

指示发送此TCP数据段的主机上用来存储传入数据段的窗口大小，也即发送者当前还可以接收的最大字节数，相当于本章前面介绍OSI/RM TPDU中的CDT（信用量分配），占16位。TCP的“窗口大小”字段与本章前面介绍的CDT一样是使用可变大小的滑动窗口协议来进行流量控制。“窗口大小”字段的值告诉接收本数据段的主机，从本数据段中所设置的“确认号”值算起，本端目前允许对端发送的字节数，是作为让对方设置其发送窗口大小的依据。假设本次所发送的数据段的“确认号”字段值501，而“窗口大小”字段值是100，则从501算起，本端还可以接收100字节（字节序号是501~600）。

（13）检验和（Checksum）

检验和是指对“数据段头”、“数据”和“伪头部”这三部分进行校验，占16位。“伪头部”包括源主机和目的主机的32位IP地址、TCP协议号（6），以及TCP数据段长度。若要了解校验和原理，可参见7.3.4节相关内容。

（14）紧急指针（Urgent Pointer）

仅当前面的URG控制位置1时才有意义，它指出本数据段中为紧急数据的字节数，占16位。“紧急指针”字段指明了紧急数据的末尾在数据段中的位置。当所有紧急数据处理完后，TCP就会告诉应用程序恢复到正常操作。要注意的一点是，即使当前窗口大小为0，也是可以发送紧急数据的，因为紧急数据无须缓存。

(15) 可选项 (Option)

“可选项”字段是可选的，且长度可变，最长可达40字节。当没有使用该字段时，TCP头部的长度是20字节。它可以包括窗口缩放选项 (Window Scale Option, WSopt)、MSS (最大数据段大小) 选项、SACK (选择性确认) 选项、时间戳 (Timestamp) 选项等。

(16) 数据 (Data)

这是由应用层的应用进程提交的数据，作为TCP数据段的“数据” (有效载荷) 部分。

10.3.3 TCP套接字

在本章前面介绍OSI/RM的传输层服务时，提到过传输通信两端的端点就是TSAP地址，对应的就是传输层协议端口。在TCP/IP网络中，同样有端点的概念，但它并不是直接采用OSI/RM中的TSAP叫法，而是称为“套接字”（Socket），就像在TCP中仍然把所传输的数据称为“数据段”，而没有采用OSI/RM中的TPDU叫法一样。当然，需要说明的是，Socket并不能直接等同于TSAP，它们只是类似，实际上Socket只是利用了TSAP地址，因为它所包括的一组参数中就有TSAP地址——端口。“套接字”最早使用于UNIX操作系统中，后来被广泛地应用于Windows和Linux系统中，成为了事实上的TCP标准。

1.套接字概述

在TCP/IP网络中，区分不同应用程序进程间的网络通信和连接时主要有以下3个参数：通信的目的IP地址、使用的传输层协议（TCP或UDP）和使用的端口号（此处说明一下，Socket不仅在TCP中有，在UDP中也同样有）。通过将这3个参数结合起来，与一个Socket进行绑定，应用层就可以与传输层一起通过套接字接口区分来自不同应用程序进程或网络连接的通信，实现数据传输的并发服务。为了区别不同的应用程序进程和连接，许多计算机操作系统为应用程序与TCP/IP交互提供了称为套接字（Socket）的接口。

Socket可以看成是在两个网络应用程序进行通信连接时的一个端点（或者称为“逻辑接口”），它是连接应用程序和网络驱动程序的桥梁，如图10-35所示。从位置上看，它与本章前面介绍的TSAP是不一样的，因为TSAP是位于传输层上边缘（但仍属于传输层），而Socket是完全位于应用层，但它调用了传输层的端口。Socket包括了TSAP地址，同样它还包括了在本章前面所介绍的服务原语，具体内容将在本节后面介绍。

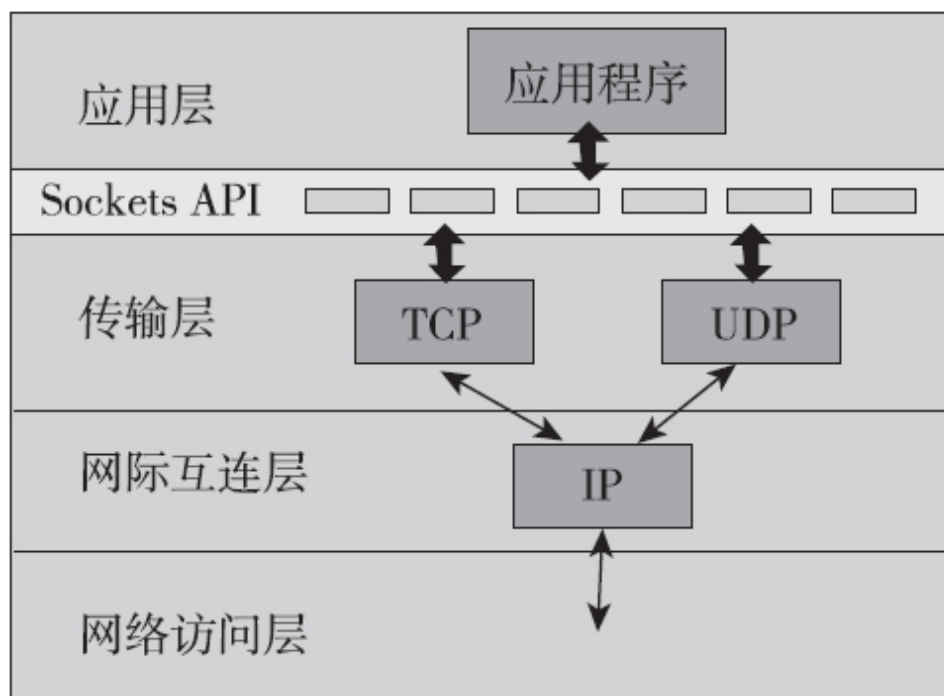


图 10-35 Socket接口的位置

在TCP/IP通信中，应用程序把数据传送给Socket，然后由Socket通过传输层向下提交给网络驱动程序并向网络上发送出去。计算机从网络上收到与该Socket绑定IP地址和端口号相关的数据后，由网络驱动程序

序通过传输层向上提交给应用层的Socket，最后应用程序从Socket中提取所要接收的数据。通信双方计算机上的网络应用程序就是这样通过Socket进行数据的发送与接收的，基本交互过程如图10-36所示。

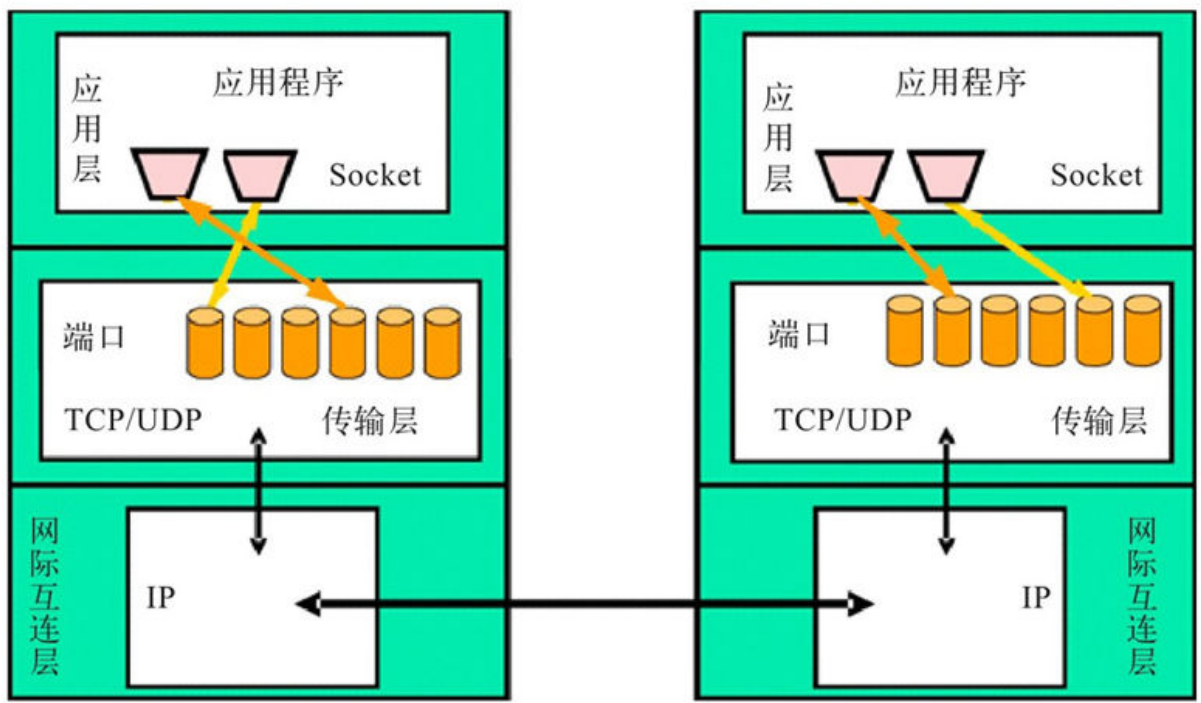


图 10-36 Socket在TCP/IP通信中的交互应用

从图10-36中可以看出，在应用层上，针对每个应用进程都有一个Socket，用来调用传输层中的一个特定端口（当然，这个端口对于常规服务来说是固定的，而对于非常规服务来说不是固定的）。也就是说，应用层的Socket和传输层的端口都有很多，但位于网络层，对于一台具体的主机和具体的网络来说，它的IP地址却是唯一的，也只有Socket和端口与IP地址之间是“多对一”的关系。

2.Socket原语

在TCP Socket中，包括了如表10-6所示的一组（共8个）TCP数据传输服务原语。表中前4个原语由服务器按照顺序执行（但并不是说服务器仅调用这4个原语）。其中，**SOCKET**原语用来创建一个新的端点，并且在传输实体中为它们分配相应的空间。**SOCKET**原语调用成功后，返回一个普通的文件描述符，以便在后续其他的原语调用中使用。但由于新创建的套接字并没有网络地址，需要通过**BIND**原语为它进行分配，所以**BIND**原语的用途就是为新建的套接字绑定一个本地网络地址。

表 10-6 TCP Socket 原语

原语名称	含 义
SOCKET	创建一个新的通信端点
BIND	将一个本地地址关联到一个套接字上
LISTEN	通告愿意接受连接，并给出队列大小
ACCEPT	阻塞调用方，直到有人企图进行连接
CONNET	主动尝试建立一个连接
SEND	在指定的连接上发送数据
RECV	从指定的连接上接收数据
CLOSE	释放指定的连接

BIND原语执行后需要调用**LISTEN**原语，它为新建的套接字分配一定的缓存空间，以便让后面要进来的连接进行排队，这样就可以使得多个客户可以同时对一个服务器进行访问。但此时服务器还不是阻塞状态，也就是服务器仍没有进入等待连接状态，这里再需要调用一个**ACCEPT**原语。当一个请求连接的TCP数据段到来时，传输实体会通

过调用**ACCEPT**原语创建一个新的套接字，并返回一个与其关联的文件描述符。这个新的套接字与原来由**SOCKET**原语创建的套接字具有相同的属性。这时，服务器就可以调用一个进程来处理这个套接字上新的连接，而服务器本身又回去继续等待原套接字上的下一个连接。

以上介绍的是由服务器调用的4个原语，而在客户端使用的原语主要包括**SOCKET**、**CONNECT**、**SEND**、**RECV**。首先，由**SOCKET**原语创建一个新的套接字，但由于服务器并不关心它所用的网络地址，所以客户端不必调用**BIND**原语。然后，通过**CONNECT**原语阻塞客户端，并主动发起一个**TCP**传输连接。当**CONNECT**原语调用完成，即收到服务器端发来的确认数据段后，客户端解除阻塞状态，建立传输连接。到此为止，双方都可以使用**SEND**和**RECV**原语在建立起来的全双工连接上进行数据的接收或发送。

服务器和客户端都可以调用**CLOSE**原语，用来释放当前**TCP**传输连接。当双方都执行了**CLOSE**原语后，连接才会真正释放。

TCP Socket原语调用的基本流程如图10-37所示。

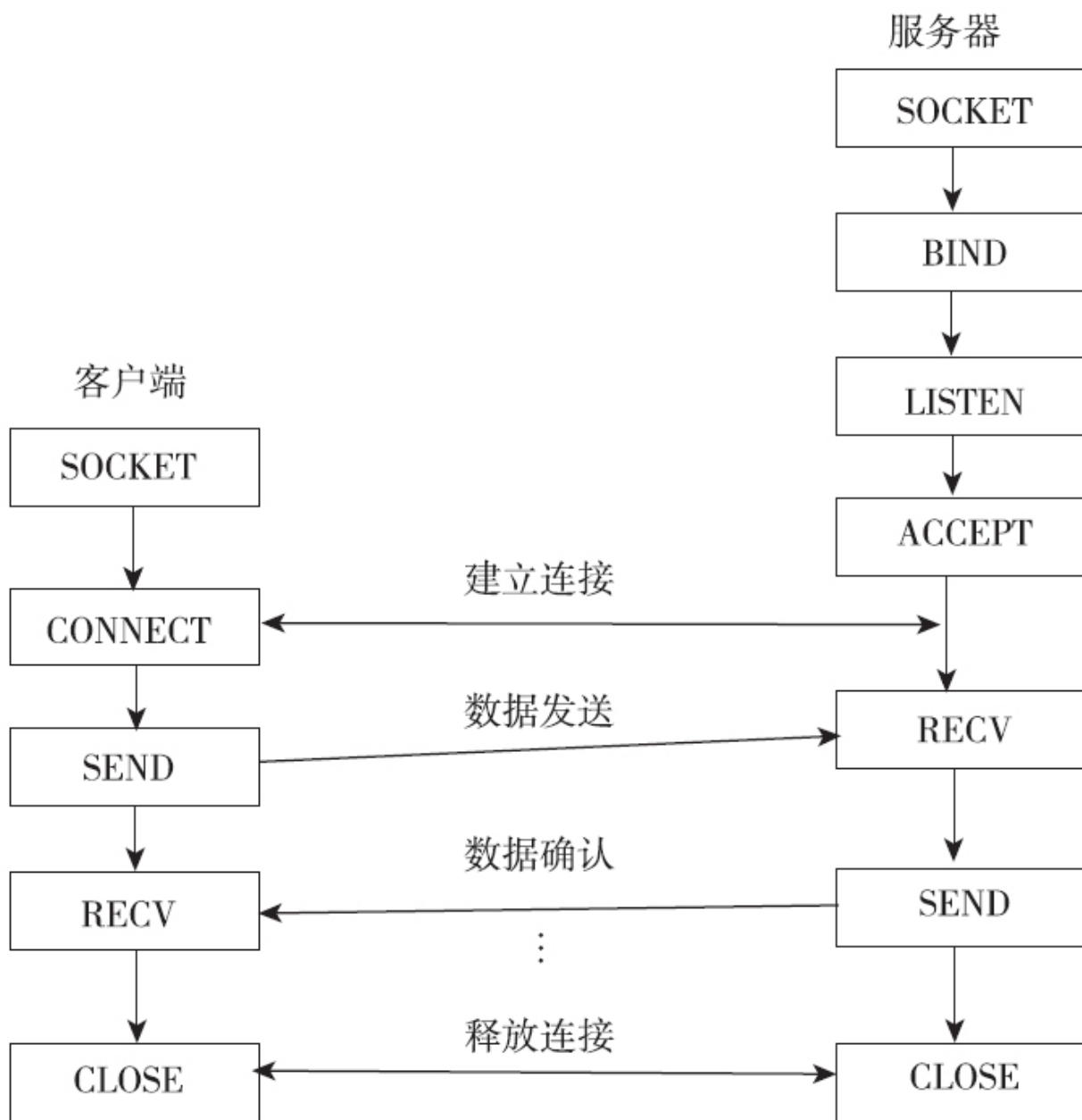


图 10-37 TCP Socket原语调用的基本流程

10.3.4 TCP端口

在上文已经提到过，传输层TSAP地址就是传输层协议端口，是对网络应用进程的一种标识。这里要区分“端口”与“接口”两个不同概念，接口是物理的，而“端口”却是一种抽象的软件结构，包括一些数据结构和I/O（输入/输出）缓冲区，故属于软件接口范畴，是传输层TSAP地址。

每个端口都拥有一个称为“端口号”的整数描述符，用来标识不同的端口或进程。在TCP数据段格式中，分别定义一个16比特长度的“源端口”和“目的端口”两个字段，也就是说，可定义 2^{16} 个端口，其端口号为 $0 \sim 2^{16} - 1$ （65535）。由于TCP/IP传输层中的TCP和UDP是两个完全独立的软件模块，因此各自的端口号也相互独立，即各自可独立拥有65535个端口。每种应用层服务或应用进程都具有与传输层唯一连接的端口，并且使用唯一的端口号进行区分。

可以将这65535个TCP端口分为以下三大类（它们是由IANA管理的）。

（1）保留端口

通常将0~1023号TCP端口保留，因此，这类端口也称为“常规端口”，或者“公认端口”（well-known port）。这些端口基本上都已固定

地分配给了已知的网络应用协议，如HTTP（对应WWW服务）中的80号端口、Telnet协议中的23号端口、SMTP中的25号端口等。目前，这一类端口的端口号分配已经被广大网络应用者接受，形成了标准，在各种网络的应用中调用这些端口号就意味着使用它们所代表的应用协议，因此不能被分配给其他网络应用协议使用。表10-7显示了部分常规服务TCP端口及所对应的服务。

表 10-7 部分常规服务 TCP 端口

端口号	对应的协议	用途说明
20	FTP	FTP 数据传输
21	FTP	FTP 控制消息
22	SSH	安全登录
23	Telnet	远程登录
25	SMTP	邮件发送
42	WINS	Internet 名称解析服务
53	DNS	域名解析
69	TFTP	小型文件传输协议
79	Finger	查询有关用户的信息
80	HTTP（或 WWW）	万维网服务
110	POP3	邮件接收服务
115	SFTP	简单文件传输协议
119	NNTP	UESNET 新闻
137	NetBIOS	NetBIOS 名称服务
138	NetBIOS	NetBIOS 数据报服务
139	NetBIOS	NetBIOS 会话服务
143	IMAP	邮件消息管理
500	ISAKMP	Internet 安全关联和密钥管理
520	RIP	基于距离矢量的动态路由

(2) 动态分配端口

动态分配端口的端口号一般都大于**1024**，它们没有分配给固定的网络应用服务，因此，可以动态地分配给任意网络服务应用程序使用。也就是说，在使用网络应用软件访问网络时，网络应用软件可以向系统申请一个大于**1024**的端口临时代表这个软件与传输层交换数据，并且使用这个临时的端口与网络上的其他主机通信。

注意，动态分配端口中有许多是被木马类的黑客程序所利用的，通过查看某个端口是否被占用就可以了解当前系统中是否中了某一种木马程序。

(3) 注册端口

注册端口比较特殊，它也是固定为某个应用服务的端口，但是它所代表的不是已经形成标准的应用层协议，而是某个软件厂商开发的应用程序，如CCProxy中的8080号端口。这类注册端口的端口号一般也都大于**1024**。

当网络中的两台主机进行通信时，为了表明数据是由源端的哪一种应用发出的，以及数据所要访问的是目的端的哪一种服务，TCP/IP会在传输层封装数据段时，把发出数据的应用程序的端口作为源端口，把接收数据的应用程序的端口作为目的端口，添加到数据段的头部中，从而使主机能够同时维持多个会话的连接，使不同应用程序的数据不会发生混淆。

10.3.5 TCP连接的状态转移

从表10-6中可以看出，TCP Socket服务原语只有8个，比OSI/RM体系结构中定义的传输层服务原语还要少，但是在这8种TCP Socket服务原语中，有的原语又可以有不同的状态，如表10-8所示。我们把在TCP传输连接的建立和释放过程中的通信双方主机的这些状态称为“有限状态机”（Finite State Machine，FSM）。下面具体阐述一下在TCP传输连接建立、释放过程中的这些原语状态的变化。

表 10-8 TCP 连接状态

状 态	描 述
CLOSED	呈阻塞、关闭状态，表示主机当前没有活动的传输连接或正在进行传输连接
LISTEN	呈监听状态，表示服务器正在等待新的传输连接进入
SYN RCVD	表示主机已收到一个传输连接请求，但尚未确认
SYN SENT	表示主机已经发出一个传输连接请求，等待对方确认
ESTABLISHED	传输连接建立，通信双方进入正常数据传输状态
FIN WAIT 1	（主动关闭）主机已经发送关闭连接请求，等待对方确认
FIN WAIT 2	（主动关闭）主机已收到对方关闭传输连接确认，等待对方发送关闭传输连接请求
TIMED WAIT	完成双向传输连接关闭，等待所有分组消失
CLOSING	双方同时尝试关闭传输连接，等待对方确认
CLOSE WAIT	（被动关闭）收到对方发来的关闭传输连接请求，并已确认
LAST ACK	（被动关闭）等待最后一个关闭传输连接确认，并等待所有分组消失

图10-38描述了TCP通信主机在传输连接建立和释放过程中的各种有限状态机，方框中表示的是通信主机在不同时期的状态，箭头表示状态之间的转换，旁边的注释表示状态转换过程中所需进行的动作（包括调用Socket服务原语以及TCP数据段的发送和接收等）。图10-

38中用粗线表示客户端主动和被动的服务器端连接建立的正常过程，其中客户端的状态转移用带箭头的粗实线表示，服务器端的状态转换用带箭头的粗虚线表示。带箭头的细线表示一些不常见的事件，如复位、同时打开、同时关闭等。

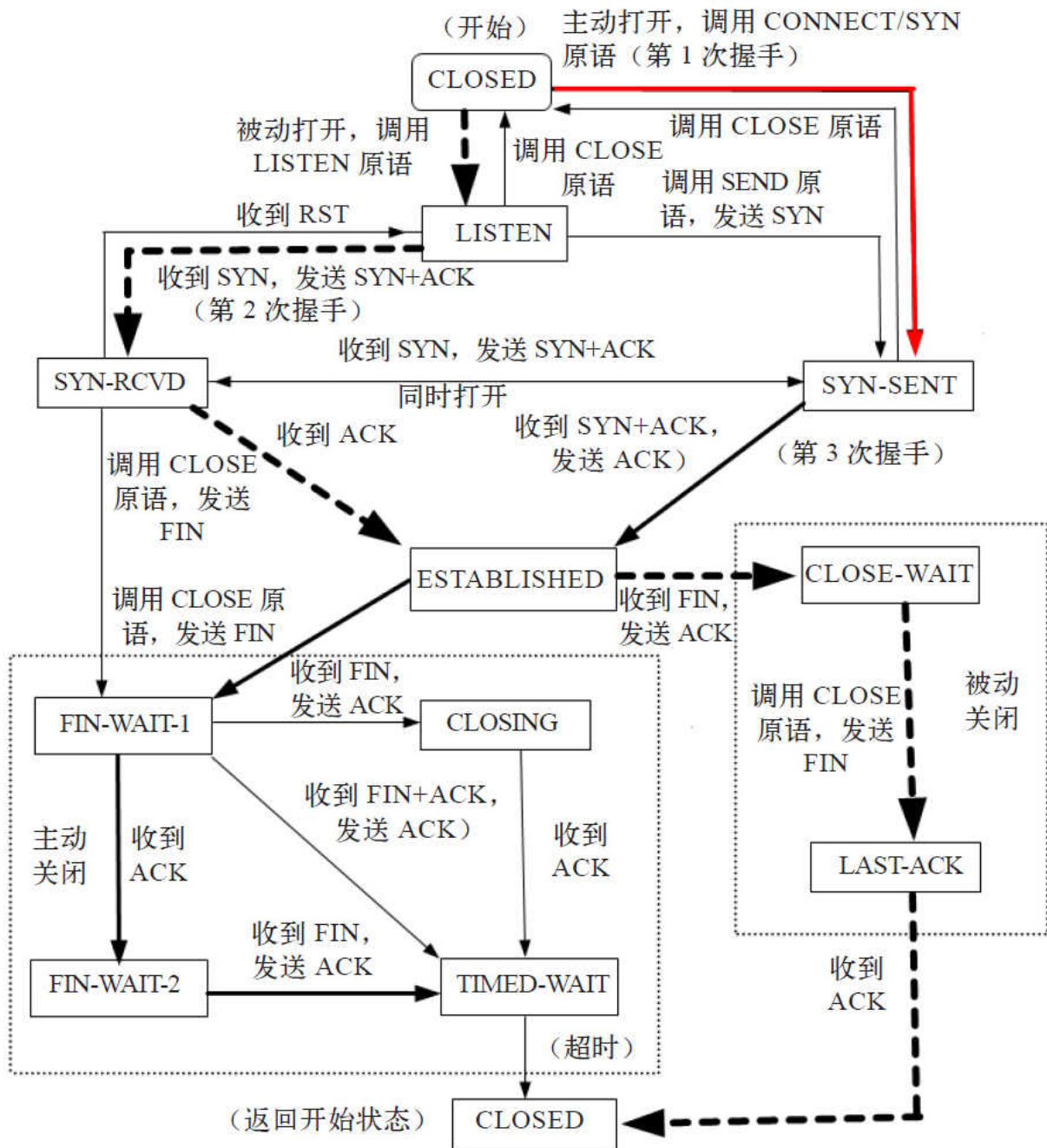


图 10-38 TCP传输连接有限状态机转换流程

每个连接均开始于CLOSED状态。当一方执行了被动的连接原语（LISTEN）或主动的连接原语（CONNECT）时，它便会离开CLOSED状态。如果此时另一方执行了相对应的连接原语，连接便建立了，并且状态变为ESTABLISHED。任何一方均可以首先请求释放连接，当连接被释放后，状态又回到了CLOSED。

为了让读者看清楚客户端和服务端各自的状态转移流程，我们先沿着带箭头的粗实线路径来看客户端的状态转移过程，然后再沿着带箭头的虚实线路径来看服务器的状态转移过程。

- 1) 一开始，服务器应用层首先调用LISTEN原语从CLOSED状态进入被动打开状态（LISTEN），等待客户端的连接。

- 2) 当客户端的一个应用程序调用CONNECT原语后，本地的TCP实体为其创建一个连接记录并标记为SYN SENT状态，然后给服务器发送一个SYN数据段（SYN字段置1）。这是TCP传输连接的第一次握手。

- 3) 服务器在收到一个客户端的SYN数据段后，其TCP实体向客户端发送确认ACK数据段（ACK字段置1），同时发送一个SYN数据段（SYN字段置1，表示接受同步请求），进入SYN RCVD状态。这是TCP传输连接的第二握手。

说明 这里可能有一个非正常事件发生，那就是如果此时服务器不想建立传输连接，那么由其应用层调用**CLOSE**原语，向客户端发送一个**FIN**数据段（**FIN**字段置1），然后进入**FIN WAIT 1**状态，等待客户端确认。当客户端收到服务器发来的**FIN**数据段后，向服务器发送一个**ACK**确认数据段后进入到**CLOSING**状态，表示双方同时尝试关闭传输连接，等待对方确认。在服务器收到客户端发来的**ACK**数据段后即进入到**TIMED WAIT**状态，在超时后双方即关闭连接。这是一种突发、非正常的连接关闭事件。

4) 客户端在收到服务器发来的**SYN**和**ACK**数据段后，其**TCP**实体给服务器端发送一个**ACK**数据段，并进入**ESTABLISHED**状态。这是**TCP**连接的第三次握手。

5) 服务器在收到来自客户端的**ACK**确认数据段后，完成整个**TCP**传输连接的全部三次握手过程，也进入**ESTABLISHED**状态。

此时，双方可以自由进行数据传输了。当一个应用程序完成数据传输任务后，它需要关闭**TCP**连接。假设仍由客户端发起主动关闭连接。

6) 客户端应用层调用**CLOSE**原语，本地的**TCP**实体发送一个**FIN**数据段（**FIN**字段置1），并等待服务器的确认响应，进入**FIN WAIT 1**状态。

说明 这里又可能有一个非正常的事件发生，那就是客户端在FIN WAIT 1状态收到服务器的FIN和ACK数据段后（而不是像从FIN WAIT 2状态进入那样只收到服务器的ACK确认数据段），向服务器发送一个ACK数据段，直接就进入到TIMED WAIT状态。在超时后双方即关闭连接。

7) 服务器在收到来自客户端的FIN数据段后，它给客户端返回一个ACK数据段（ACK字段置1），进入CLOSE WAIT状态。

8) 客户端在收到来自服务器的ACK确认数据段后，进入FIN WAIT 2状态，此时连接在一个方向上就断开了，但仍可以接收服务器端发来的数据段。

9) 当服务器收到客户端发来的FIN数据段时就知道客户端已有数据发送了，在本端已接收完全部的数据后，也由应用层调用CLOSE原语，请求关闭另一个方向的连接，其本地TCP实体向客户端发送一个FIN数据段，并进入LAST ACK状态，等待最后一个ACK确认数据段。

10) 在客户端收到来自服务器的FIN数据段后，向服务器发送最后一个ACK确认数据段，进入TIMED WAIT状态。此时，双方连接均已经断开，但TCP实体仍要等待一个2倍数据段MSL（Maximum Segment Lifetime，最大数据段生存时间），以确保该连接的所有分组全部消

失，防止出现确认丢失的情况。当定时器超时后，TCP删除该连接记录，返回到初始状态（CLOSED）。

11) 服务器收到客户端最后一个ACK确认数据段后，其TCP实体便释放该连接，并删除连接记录，也返回到初始状态（CLOSED）。

10.3.6 TCP传输连接的建立

TCP是一个面向连接的传输层协议，因此，无论哪一方向的另一方发送数据，都必须先在双方之间建立一条传输连接。本节将详细讨论一个TCP传输连接是如何建立的。

1. 单方主动连接的TCP连接建立过程

TCP/IP体系结构中的TCP也是使用三次握手机制来建立传输连接的，这与在本章前面介绍的OSI/RM体系结构中传输层为了避免重复连接而采取的三次握手机制是一样的，具体流程如图10-39所示。其实，整体过程在图10-38中有全面的体现，这里只是单独把TCP传输连接建立过程列出来而已。具体步骤如下：

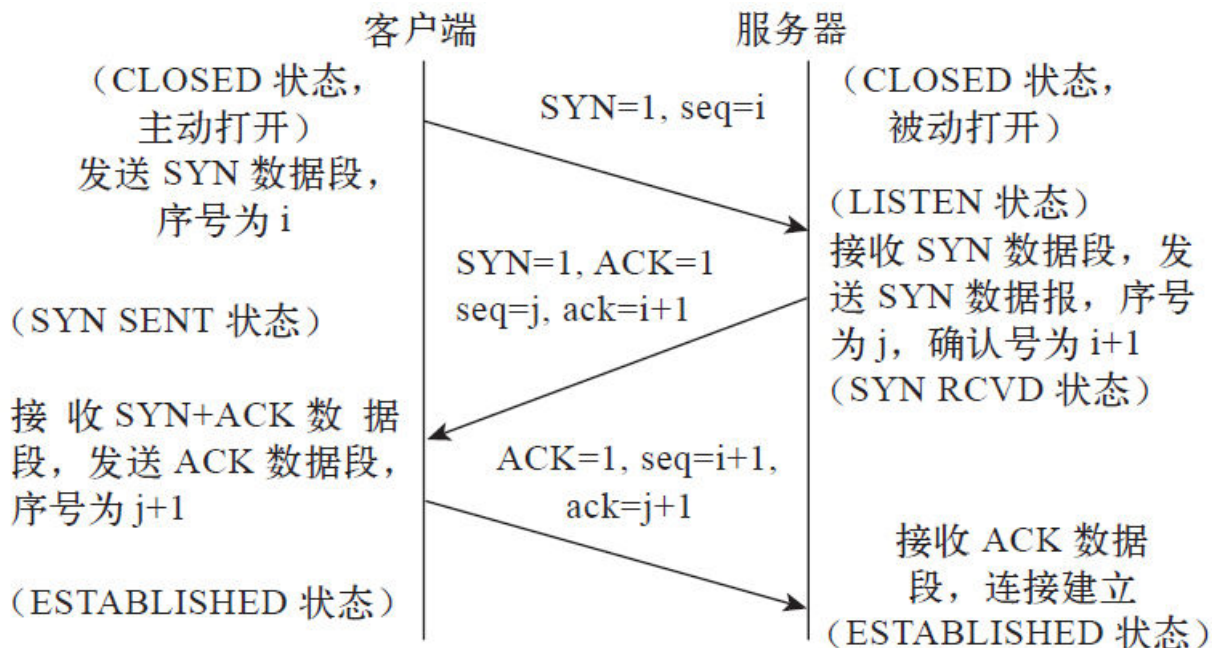


图 10-39 TCP传输连接建立的三次握手过程

1) 首先是服务器初始化的过程，从CLOSED（关闭）状态开始通过顺序调用SOCKET、BIND、LISTEN和ACCEPT原语创建Socket套接字，进入LISTEN（监听）状态，等待客户端的TCP传输连接请求。

2) 客户端最开始也是从CLOSED状态开始调用SOCKET原语创建新的Socket套接字，然后在需要再调用CONNECT原语时，向服务器发送一个将SYN字段置1（表示此为同步数据段）的数据段（假设初始序号为i），主动打开端口，进入SYN SENT（已发送连接请求，等待对方确认）状态。

3) 服务器在收到来自客户端的SYN数据段后，返回一个SYN字段置1（表示此为同步数据段）、ACK字段置1（表示此为确认数据

段)、ack (确认号) = $i+1$ 的应答数据段 (假设初始序号为 j)，被动打开端口，进入到 **SYN RCVD** (已收到一个连接请求，但未进行确认) 状态。这里要注意的是，确认号是 $i+1$ ，而不是 i ，表示服务器希望接收的下一数据段序号为 $i+1$ 。

4) 客户端在收到来自服务器的 **SYN+ACK** 数据段后，向服务器发送一个 **ACK=1** (表示此为确认数据段)、序号为 $i+1$ 、ack = $j+1$ 的确认数据段，同时进入 **ESTABLISHED** (连接建立) 状态，建立单向连接。要注意的是，此时序号为 $i+1$ ，确认号为 $j+1$ ，表示客户端希望收到服务器的下一个数据段的序号为 $j+1$ 。

5) 服务器在收到客户端的 **ACK** 数据段后，进入 **ESTABLISHED** 状态，完成双向连接的建立。

连接可以由任一方或双方发起，一旦连接建立，数据就可以双向对等地流动，而没有所谓的主从关系。三次握手是连接两端正确同步的充分必要条件，因为 **TCP** 建立在不可靠的分组传输服务之上，报文可能丢失、延迟、重复和乱序，因此协议必须使用超时和重传机制。如果重传的连接请求和原先的连接请求在连接正在建立时到达，或者当一个连接已经建立、使用和结束之后，某个延迟的连接请求才到达，就会出现问题。采用三次握手协议可以解决上述这些问题。如客户端发送的 **ACK** 数据段就是为了避免因网络延迟而导致的重复连接，因为

这时客户端通过检查ACK数据段中的确认号就可得知该连接请求是否已失效。

经验之谈 对比图10-39和图10-21，可以看出，TCP传输连接的建立与OSI/RM体系结构中TCP传输协议的传输连接建立过程既存在相似之处（如三次握手机制），又有较大区别，主要表现在：①在OSI/RM体系结构中，只有DT TPDU和ED TPDU才有序列号，在返回的确认类TPDU中是没有TPDU序号的，因此，在图10-21中并没有标注确认TPDU的序列号，而在TCP传输连接中，每个数据段（无论是否携带数据）都有序列号，都需要标示其对应的序列号；②在OSI/RM体系结构的传输协议中，因为不同服务原语可以使用不同的特定类型TPDU，所以在OSI/RM体系结构的TPDU中不存在像TCP数据段中的ACK、SYN等类型的控制位。在TCP中，所有数据段格式都是一样的，不同的只是不同类型的TCP数据段的这些字段的取值不同；③TCP传输连接中使用“确认号”字段与OSI/RM TPDU中的“YR-TU-NR”（你的TPDU序列号）字段的功能是完全一样的，都是显示对端希望接收的下一个数据段序列号，暗示该号码前面的所有数据段均已正确接收。

2.双方同时主动连接的TCP连接建立过程

正常情况下，传输连接都是由一方主动发起的，但也有可能双方同时主动发起连接，此时就会发生连接碰撞，最终只有一个连接能够建立起来。所有连接都是由它们的端点进行标识的，如果第一个连接

请求建立起一个由套接字 (x,y) 标识的连接，而第二个连接也建立了这样一个连接，那么在TCP实体内部只有一个套接字表项。

当出现同时发出连接请求时，两端几乎在同时发送一个SYN字段置1的数据段，并进入SYN_SENT状态。当每一端收到SYN数据段时，状态变为SYN_RCVD，同时它们都再发送SYN字段置1、ACK字段置1的数据段，对收到的SYN数据段进行确认。当双方都收到对方的SYN+ACK数据段后，便都进入ESTABLISHED状态。图10-40显示了这种同时发起连接的连接过程，但最终建立的是一个TCP连接，而不是两个，这点要特别注意。

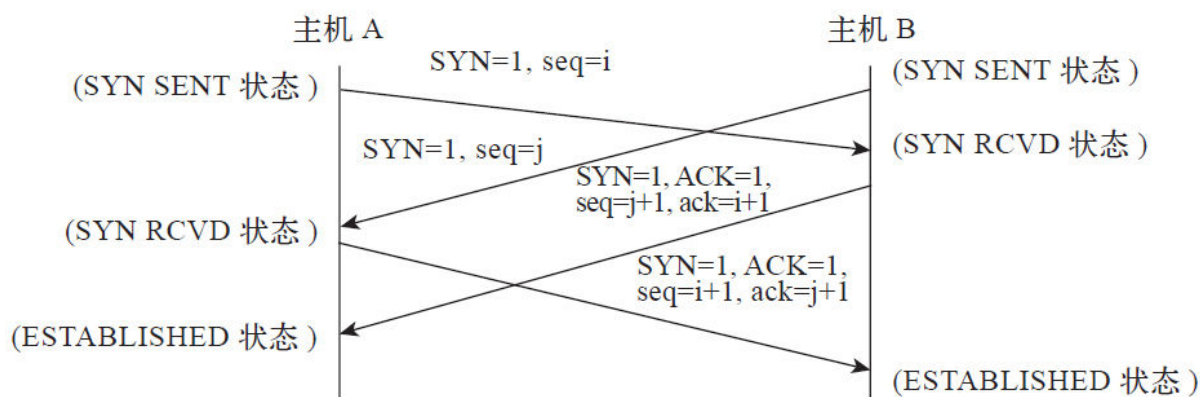


图 10-40 同时发起连接的TCP连接建立过程

从图10-40中可以看出，一个双方同时打开的传输连接需要交换4个数据段，比正常的传输连接建立所进行的三次握手多交换一个数据段。此外，要注意的是，此时没有将任何一端称为客户端或服务器，因为每一端既是客户端又是服务器。

10.3.7 TCP传输连接的释放

当TCP连接建立起来后，就可以在两个方向传送数据流。当TCP的网络应用进程再没有数据需要发送时，就可以发出关闭连接命令，释放连接。TCP是通过发送FIN字段置1的数据段来作为关闭传输连接的命令，从而关闭本端数据流的，但是本端仍然还可以继续接收来自对端的数据，直到对端也使用了同样的方法关闭那个方向的数据流为止，这时整个双方传输连接就彻底关闭了。

1. 单方主动关闭的TCP连接释放过程

相对TCP传输连接建立的三次握手过程来说，TCP传输连接的释放过程要稍微复杂一些，需要经过四次握手过程。这是由TCP的半关闭（half-close）特性造成的，因为这一个TCP连接是全双工（即数据在两个方向上能同时传递），每个方向必须单独进行关闭。TCP传输连接关闭的原则如下：当一端完成它的数据发送任务后就可以发送一个FIN字段置1的数据段来终止这个方向的数据发送；当另一端收到这个FIN数据段后，必须通知它的应用层“对端已经终止了那个方向的数据传送”。而FIN数据段的发送是由应用层调用CLOSE服务原语的结果。TCP连接释放的四次握手过程如图10-41所示，具体描述如下：

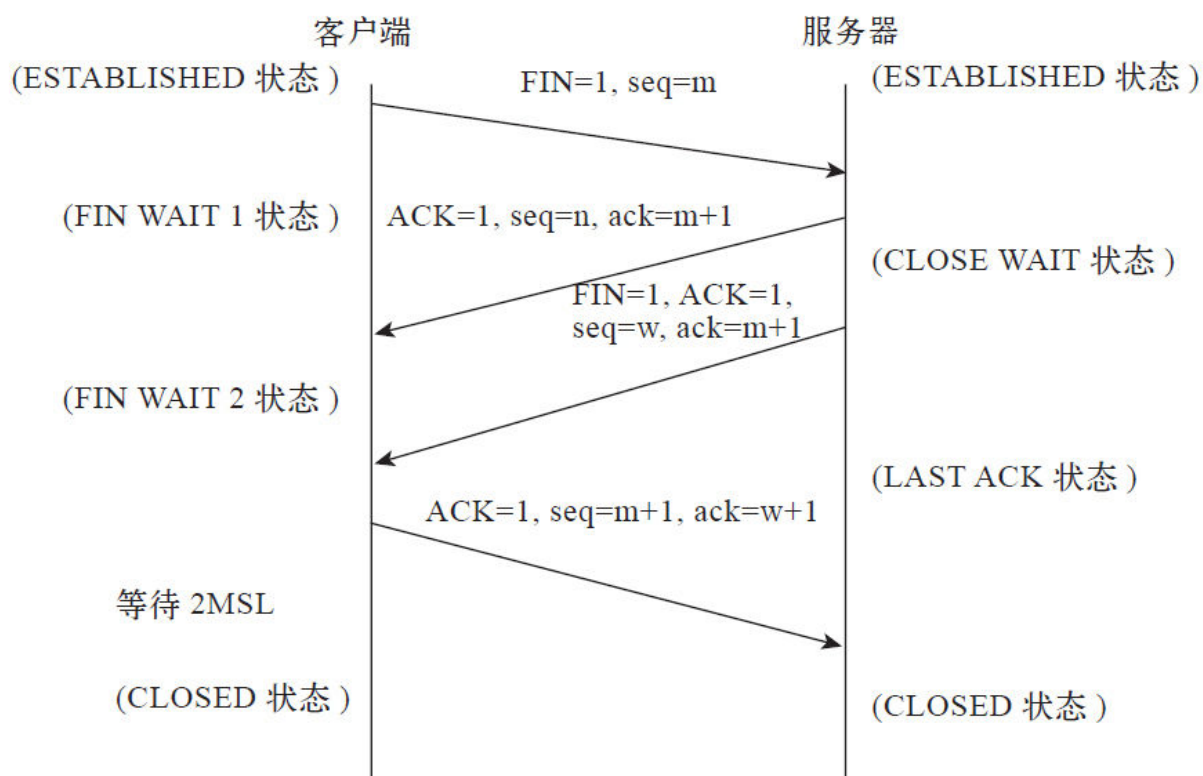


图 10-41 TCP传输连接释放的四次握手过程

1) 一开始，通信双方都处于ESTABLISHED（连接建立）状态。如果客户端认为数据全部发送完了，想结束本次传输连接，则由应用层的对应应用进程调用CLOSE服务原语，然后向服务器发出一个FIN字段置1的数据段（假设此数据段的序号为m），客户端进入FIN WAIT 1 状态，等待服务器的确认。

2) 服务器在收到客户端发来的FIN数据段后，确认客户端没有新的数据要发送了，向客户端发送一个ACK字段置1、确认号为m+1的数据段（假设此数据段序号为w，服务器与客户端的数据段序号可以不一样），表示前面的数据已全部收到了，然后进入CLOSE WAIT（关闭

等待) 状态。与此同时, 服务器的TCP实体通知对应的应用层进程, 释放从客户端到服务器方向的传输连接, 进入半关闭状态。但此时服务器仍可以向客户端发送数据段, 客户端也可接收来自服务器的数据, 而且这可能要持续一段时间, 直到服务器的数据也全部发送完为止。

3) 当客户端收到服务器的ACK数据段后, 进入到FIN WAIT 2状态, 进一步等待服务器发出连接释放的数据段。

4) 当服务器发送完全部的数据后, 其对应的应用进程也会通知TCP实体释放此方向的TCP传输连接, 向客户端发送FIN字段置1、ACK字段置1、ack=m+1 (假设此时的数据段序号已变为w) 的确认数据段。这时服务器进入LAST ACK (最后确认) 状态, 等待客户端的确认。

5) 客户端在收到服务器的FIN+ACK数据段后, 向服务器发送一个ACK字段置1、ack=w+1、序号为m+1的数据段, 进入到TIME WAIT状态。但此时TCP连接还没有释放, 必须等待2MSL时间 (RFC793建议设置MSL为2分钟) 后, 客户端才进入到CLOSED状态, 彻底释放了TCP连接。

6) 服务器在收到客户端发来的ACK数据段后, 也进入CLOSED状态, 彻底释放连接。此时, 已经完成整个TCP传输连接释放过程。

2.双方主动关闭的TCP连接释放流程

与可以双方同时建立TCP传输连接一样，TCP传输连接关闭也可以由双方同时主动进行（正常情况下都是由一方发送第一个FIN数据段进行主动连接关闭，另一方被动接受连接关闭），如图10-42所示。具体描述如下：

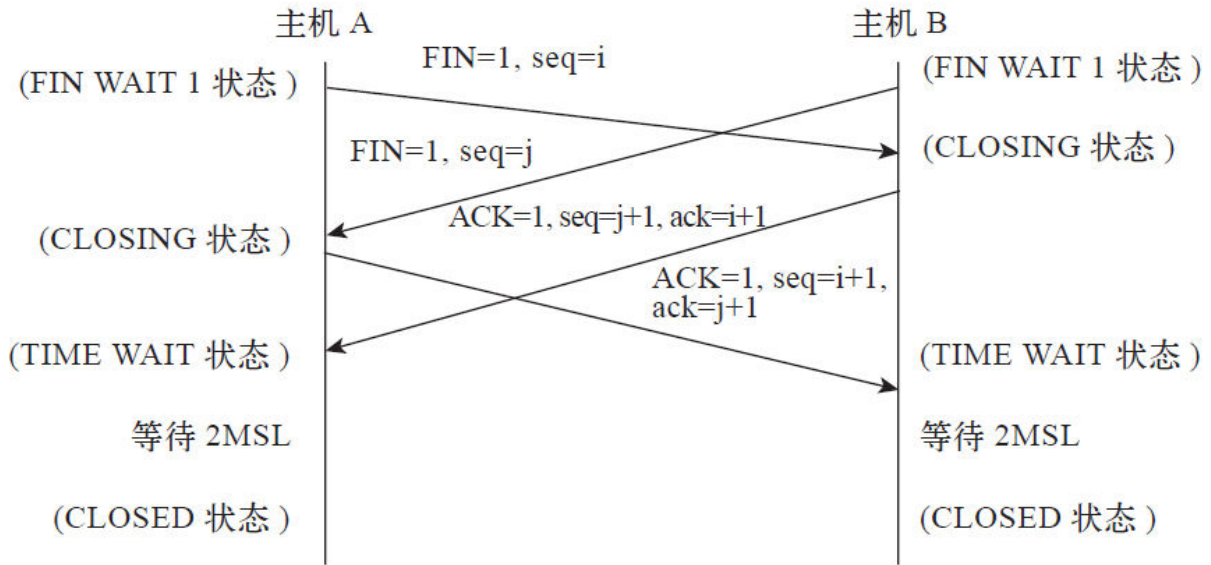


图 10-42 同时主动关闭TCP连接的流程

当两端对应的网络应用层进程同时调用CLOSE原语，并发送FIN数据段执行关闭命令时，两端均从ESTABLISHED状态转变为FIN WAIT 1状态。任意一端收到对端发来的FIN数据段后，其状态均由FIN WAIT 1转变到CLOSING状态，并发送最后的ACK数据段。当收到最后的ACK数据段后，状态转变为TIME_WAIT，在等待2MSL后进入到CLOSED状态，最终释放整个TCP传输连接。

10.4 TCP的可靠传输

本章前面一直在说TCP是一个可以提供可靠数据传输的传输层协议，那么它到底是如何来保障可靠传输的呢？下面进行具体的分析。

在TCP可靠传输方面，主要采用以下4种机制：

□一是“字节编号机制”。TCP数据段以字节为单位对数据段中的“数据”部分进行一一编号，确保每个字节的数据都可以有序传送和接收。

□二是“数据段确认机制”。TCP要求每接收一个数据段都必须由接收端向发送端返回一个确认数据段（可以用一个确认数据段一次确认前面多个数据段），其中的“确认号”表明了接收端已正确接收的数据段序号（“确认号”前面的所有数据段）。

□三是“超时重传机制”。在TCP中有一个重传定时器（Retransmission Timer, RTT），在发送一个数据段的同时也启动了该定时器。如果在定时器过期之前该数据段还没有被对方确认的话，则定时器停止，然后重传对应序号的数据段。

□四是“选择性确认（Selective ACK, SACK）机制”。在SACK支持下，仅可以重传缺少部分的数据，而不会重传那些已正确接收的数

据。

以上所说的“字节编号机制”比较好理解，因为是按字节进行编号的，所以接收端根据所接收到的数据段中的序号就可知道前面是否还有数据没接收到，数据可以按顺序向应用进程提交，在对经过了数据段的数据进行重组时也可以根据这个序号进行正确的重组。本节重点介绍后面3种机制。

10.4.1 TCP的数据段确认机制

1.几个重要概念的理解

为了更好地理解TCP的确认机制，下面回顾一下与TCP确认机制密切相关的一些重要概念。

(1) 数据段

数据段指TCP对从应用层接收的数据进行分割所得到的数据块（也可以是没有经过分割的整个数据报文，只要它的大小在MSS之内），通常包括千个以上字节，而且必须是整数倍字节数。正因如此，TCP发送的是字节流，而不是通常所说的报文流，因为在TCP数据段中没有报文边界。

(2) 序号

TCP发送的数据段中“数据”部分（不包括TCP数据段头部），每个字节都有一个序号，每个数据段中的“序号”字段是以该数据段中第一个字节的序号进行填充的。

（3）窗口大小

窗口大小是本端要告诉对端当前可以接收的数据量，也暗示着对端此时可以一次性发送的数据大小，以字节为单位，但这里仅是针对数据段中的“数据”部分，不包括TCP数据段头部，因为数据段到了对端的应用层后仅提交“数据”部分。TCP可以一次性连续发送窗口大小的数据（但实际上发送数据的大小还会受当前可用的“发送窗口大小”影响），其中包括一个或多个TCP数据段。但窗口大小必须小于对应网络中MTU（最大传输单元）值大小，否则到了数据链路层还是要进行数据分割的。同时要注意的是，“窗口大小”字段是随着接收端可用“接收窗口大小”变化而变化的，不是固定的。

注意 无论是发送端，还是接收端，都分别有“发送窗口”和“接收窗口”这两个窗口，其大小分别为用于发送数据和接收数据的缓存大小。这两个缓存的物理大小对于具体的主机来说是固定不变的，除非人为扩展物理内存，否则不会再扩大，只能沿着缩小的方向进行变化。

（4）确认号

确认号指发送包含这个“确认号”的数据段的一端期望接收另一端的下一个数据段的起始序号。同时也暗示了在此序号前的所有字节数据均已正确接收。

(5) ACK

ACK是一个表明“确认号”字段是否有效的标志位。只有ACK字段的值为1，数据段中的“确认号”才有意义，否则数据段中的“确认号”没有意义，即不具有上面所说的“确认号”含义，因为TCP通常不会针对单个数据段进行确认，而是一次性对多个连续数据段进行确认。只需要对最近收到的那个数据段进行确认，即表明前面所有数据段均已正确接收。

2.TCP确认机制的特性

除了要理解以上几个重要概念外，还需要对TCP确认机制的以下重要特性进行正确理解。

(1) TCP可一次连续发送多个数据段

TCP不需要等待接收对方发送的确认数据段（“ACK”字段为1的数据段）就可以一次性连续发送多个数据段，这样可大大提高数据发送效率。但一次性可发送多少个数据段是受对方返回的“窗口大小”字段

值和当前可用“发送窗口”大小双重限制的。因为发送端对还没有收到确认的数据段要进行缓存，这需要占用一定的“发送窗口”大小。

假设发送端的物理“发送窗口”和接收端的物理“接收窗口”大小均为2000字节，设每个数据段大小为100字节，而接收端返回的数据段中显示“窗口大小”字段值也为2000，同时发送端此时的“发送窗口”还有两个数据段（200字节）的数据没收到确认，则此时发送端可发送的数据段数量为18（ $2000/100-2$ ），即1800字节，不能发送全部的2000字节数据。

如果接收端返回的数据段中显示的“窗口大小”值为1000，则此时发送端可以1000字节数据，因为所发送的1000字节数据，再加上原来还没收到确认的200字节数据，小于发送端的物理“发送窗口”大小——2000。

（2）仅对连续接收的数据段进行确认

返回的确认数据段中的“确认号”字段值仅代表对端已正确接收的连续数据段（最高字节序号+1），而不一定是已正确接收数据段中的“最高序号+1”，因为中间可能还有些数据段因为网络延迟而暂时未收到，或出现了传输错误而丢失了。

假设每个数据段的长度大小均100字节，接收端收到了序号为1、101、201、401四个数据段。其中序号为301的数据段暂时没收到，此

时接收端返回的确认数据段中的“确认号”只能是301，而不会是501，也就是只对前三个数据段进行确认，不会对后面的401数据段进行确认，因为中间的301数据段还没收到。当后面收到了301数据段后，可能会返回一个“确认号”为501的数据段，这时就代表301和401数据段均已正确接收了。

(3) 不连续序号的数据将先缓存

当主机接收到的数据段序号不连续时，不连续部分不会向应用层的应用进程进行提交，而是先缓存在“接收窗口”中，等待接收到中断的序号的数据段后再一起提交。这时，尽管接收端已正确接收了某些数据，但仍不能及时向应用层提交，需要占用“接收窗口”空间。对于没有按先后次序正确接收的数据，在向应用层提交时会重新按数据段序号进行组合，然后再提交给应用层。

10.4.2 TCP的超时重传机制

“超时重传”是TCP保证数据可靠性的另一个重要机制，其原理是在发送某一个数据段以后就开启一个超时重传计时器（Retransmission Timer, RTT）。如果在这个定时器时间内没有收到来自对方的某个数据段的确认，发送端启动重传机制，重新发送对应的数据段，直到发送成功为止。需要注意的是，并不是RTT定时器一到，就会立即重传数据，毕竟从“发送窗口”缓存中找到对应的数据段，然后安排重新发送都是需要时间的，实际上，超时重发的时间间隔（Retransmission Time Out）要大于RTT值。

1.SRTT的计算

这里涉及两个重要问题：一是如何确定RTT值，另一个是如何计算RTO值。表面上看起来RTT值容易确定，因为它就是一个数据段往返发送端和接收端的时间总和，但在TCP通信中，中间可能经过了多个性能不一样的网络，而且不同时刻网络的拥塞程度可能不一样，这就造成不同数据段的RTT时间并不一致，甚至波动非常大。但TCP必须适应这种情况，必须能动态地跟踪这些变化并相应地改变其超时重传时间。

正因为RTT值不是固定的，所以就出现了平滑RTT（SRTT）的概念，就是在充分考虑历史RTT值的情况下所设计的一个RTT值计算公式。在最初的RFC793中，SRTT的计算公式如下：

$$\text{SRTT (新的SRTT)} = a\text{SRTT (旧的SRTT值)} + (1-a) \text{RTT (新的RTT样本值)}$$

SRTT的初始值就是第一个RTT值。这里的a是一个平滑因子，它决定了旧的SRTT值所占的权重， $0 \leq a < 1$ 。RFC2988推荐的a典型值为0.125。如果a很接近0，则表示新的SRTT值与旧的SRTT值很接近，变化不大，相反，则表示变化比较大。显然，用这种方法计算得出的各个时刻的SRTT值更加平滑，更加接近当时的网络环境。

2.RTO的计算

虽然可以通过公式计算出SRTT值，但是RTO的得出仍不是一件简单的事，因为到底是RTT定时器到后就重传，还是要再等待多长时间才重传，这是必须要考虑的。正常情况下，TCP使用 $b \text{ SRTT}$ 作为重传超时间隔（ $b > 1$ ），而且最初的值总为2（也就是在两倍SRTT时间后还没收到对应数据段的确认才重传该数据段）。

但经验表明，采用常数的b值不够灵活。于是，在1988年，Jacobson提出使用平均偏差作为标准偏差（就是 $b \text{ SRTT}$ ）的新估计算法，要求维护另一个被平滑的偏差RTTD。当一个确认数据段到达

时，可以得出SRTT和新的RTT样本值之间的偏差 $RTT_D = (SRTT - RTT)$ 。第一次测量时， RTT_D 为测量到的RTT样本值的一半，在以后的测量中按照以下公式进行计算：

$$RTT_D (\text{新的} RTT_D) = a RTT_D (\text{旧的} RTT_D) + (1-a) \times (SRTT - RTT)$$

这里的a可能与计算SRTT时的值相同，也可能不同，通常取值为0.25。SRTT是平滑RTT值，RTT是新的RTT样本值。虽然 RTT_D 并不能完全等同于标准偏差，但已能足够地反映SRTT值的动态变化。现在大多数TCP实现都是使用这个算法，并且将超时重传时间设置为：

$$RTO = SRTT + 4 \times RTT_D$$

这里有一个重要的问题，那就是对于重传的数据段如何计算其RTT值。假设发送端发送了一个数据段，但在重传定时器内没有收到该数据段的确认数据段，于是重新发送该数据段，可是过了一段时间，发送端又收到了该数据段的确认数据段。这时发送端就“迷糊”了，这个确认数据段到底是对原来发送的那个数据段的确认，还是对重发的数据段的确认呢？这两个数据段的序号是一样的，如果在上次发送数据段后没有再重新发送新的数据段的话，接收端返回的确认数据段中的确认号都可能一样。如果收到的确认数据段是对重传数据段的确认，但却被发送端认为是对原数据段的确认，则这样计算出的

SRTT和RTO值肯定会偏大，否则会偏小。为此，一位发现这个问题的无线电爱好者Karn提出了一个建议，就是在计算加权平均RTT时，只要数据段被重发了，就不采用其往返时间作为计算SRTT和RTO的样本，这样得出的加权平均SRTT值和RTO值就比较准确了。

10.4.3 TCP的选择性确认机制

在上面介绍的TCP重传机制中，如果在重传定时器超时后仍没收到一个数据段的确认，则可能会重传对应序号后面的所有数据段，因为后面的这些数据段均暂时不会被确认，这明显大大降低了TCP数据传输性能。

同样假设每个数据段大小为100字节，接收端已收到1、101、201、401、501这五个序号的数据段，其中301号数据段没有收到。按照前面介绍的确认机制，虽然401、501这两个序号的数据段均已收到，但因为301号这个数据段一直没收到，所以仍然不能向发送端确认。在某个时间上，如果301号数据段的重传定时器超时了，则发送端肯定会重传这个数据段，但毕竟重传的301数据段到达对端也是需要时间的。在这个301号数据段的重传过程中，401号，甚至501号数据段的重传定时器也可能超时，这时发送端可能又对401号、501号这两个数据段进行重传。这样的重传显然是不必要，也会造成接收端重复接收401号、501号这两个数据段。

为了避免这种现象的出现，在RFC3517中出现了一种称为“选择性确认”（SACK）的机制，就是在TCP数据段格式的头部“可选项”字段

中添加一个代表支持SACK的选项。但这个选项在不同的数据段中有不同的字段名称和不同的含义。

首先，必须在建立TCP传输连接时的SYN数据段中包含“SACK-Permit”（SACK允许）字段选项，表示在今后的传输中希望收到SACK选项。然后，在其他的数据段中需要包含“SACK”字段，在这个字段中，会包含本端要告诉对端已接收到的不连续数据段。原来的“确认号”字段同样有效，但此时的SACK扩展选项也仅在确认数据段（“ACK”字段值为1时）中才有效。

“SACK-Permit”字段中包括Kind（类型）和Length（长度）两个子字段，如图10-43所示。Kind字段值固定为4，占8位，表示允许使用SACK扩展确认选项；Length字段的值固定为2，占8位，表示在SYN数据段中SACK允许扩展选项占2字节。

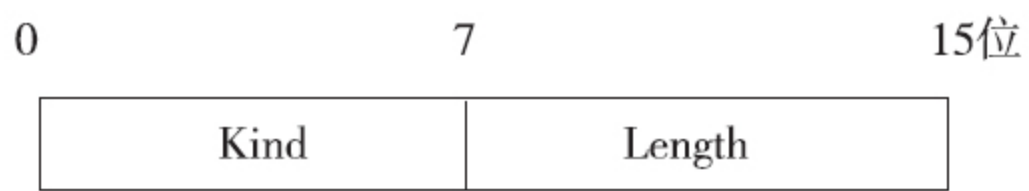


图 10-43 SYN数据段中的“SACK-Permit”选项格式

在其他非SYN数据段的“SACK”字段中，包括Kind、Length，以及各不连续数据段的起始字节号和结束字节号，如图10-44所示。Kind字段固定值为5，占8位，表示这是非SYN数据段的SACK选项；Length字

段值可变，占8位，以字节为单位表示SACK扩展选项的长度。后面是n个标识不连续数据段起始和结束序号的部分，每个序号占32位。

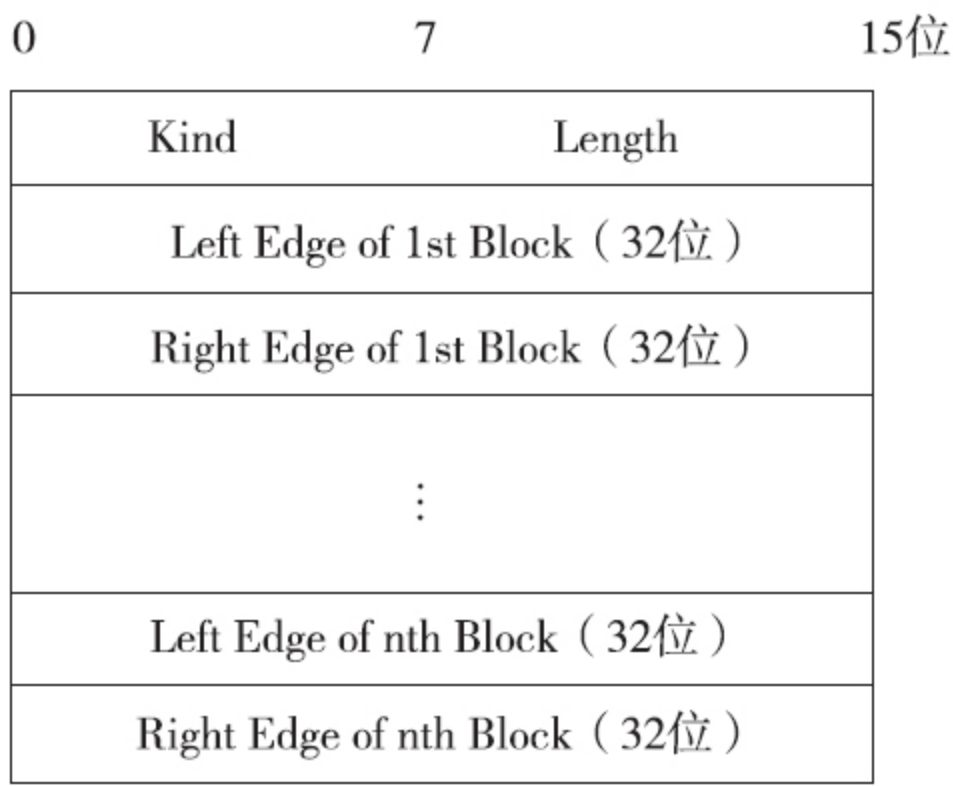


图 10-44 其他数据段中的“SACK”选项格式

由于TCP数据段头部的“可选项”字段最长为40字节，Kind和Length这两部分一共占了2字节，而表示一个不连续数据段的起始序号和结束序号要占用8字节（因为一个序号占4字节）。因此，实际上在一个数据段中最多可以标识4个不连续数据段，因为标识4个数据段的起始序号和结束序号一共要用到32字节，加上前面的Kind和Length这两部分，就有34字节了。而要标识5个数据段的起始序号和结束序号，

则还要8字节，超出了“可选项”字段40字节的限制范围。所以，在“Length”子字段中，最大值其实就是34。

发送端通过识别接收端返回到确认数据段中的SACK扩展选项就可以得知接收端已收到了哪些不连续序号的数据段，这样发送端就可以不再发送这些数据段，而只发送已丢失的数据段（发送端已发送，且在重传定时器超时后接收端仍没有收到的数据段）。

假设接收端已收到1、101、201、401、501这五个序号的数据段，在发送确认号为301的确认数据段时，在SACK扩展选项中标记401（起始序号为401，结束序号为500）和501（起始序号为501，结束序号为600）这两个不连续的数据段。这时发送端就会知道，不需要再发送401和501这两个数据段了，只需发送301号数据段即可。这样大大节省了网络资源，也提高了数据传输效率。

10.5 TCP的流量控制

虽然“流量控制”和10.6节将要介绍的“拥塞控制”是两个不同的概念，但是它们之间还是有些关联的。“流量控制”是基于通信双方的数据发送和接收速率匹配方面考虑的，其最终目的就是不要让数据发送得太快，以便接收端能够来得及接收，是一个链路两端的点对点行为。而“拥塞控制”则是基于网络中各段链路的带宽和中间设备数据处理能力方面而考虑的，不要使网络中出现数据传输阻塞，也就是不要让发送端发送的数据大于接收端数据处理能力，是一个端到端的行为。但流量控制又可能对拥塞控制有所帮助，其实解决好TCP连接所经过的所有链路的流量控制问题，也就基本上解决了拥塞控制问题了。本节先介绍TCP的流量控制。

10.5.1 TCP的流量控制简介

TCP的流量控制是采用滑动窗口协议来进行的。而在本章前面已提到过，TCP数据段是以字节为单位进行编号的，但由于一个数据段只有一个TCP头部，所以TCP是以数据段为单位进行传输的，接收端通过TCP头部来识别所接收的数据属于哪个数据段。一个数据段只要没有完全接收，接收端就不会认为已接收了该数据段，就像我们平时通过QQ等工具传输文件时，只要还有部分没接收完，则我们的计算机就不会

认为该文件已正确接收，即使已有了文件名，也打不开已接收的部分。

1.正常情况下的滑动窗口流量控制机制

在本章前面已提到过，在通信双方主机上都分别有一个“发送窗口”和一个“接收窗口”。其窗口大小又分为“物理窗口大小”和“可用窗口大小”两种。对于一台具体的主机来说，“物理窗口大小”值是固定的，要视对应主机所配置的缓存大小而定；而“可用窗口大小”值是可变的，一端会根据另一端的“接收窗口”大小（会在每个数据段的“窗口大小”字段中设置）调整本端的“发送窗口大小”，也就是说，TCP的可用“发送窗口大小”是在不断变化的，而不是固定的，这也说明了TCP每次发送的数据段数（即数据大小）可能都不一样。而可用“接收窗口”大小也会因向已发送确认的数据段数量和所接收到的不连续序号数据段数量（因为已接收但不是连续序号的数据段是仍需要缓存在物理“接收窗口”中的）而在不断变化。下面以一个具体的示例来介绍“TCP滑动窗口机制”。

图10-45是一个滑动窗口示例，其中的序号为每个数据段的序号，也就是对应数据段的第一个字节序号。现假设每个字段的大小为100字节（这仅是为了方便介绍，实际的数据段大小一般都在千个字节以上），物理“发送窗口”大小为500字节（这也是为了方便介绍，实际的窗口大小远大于这个值）。

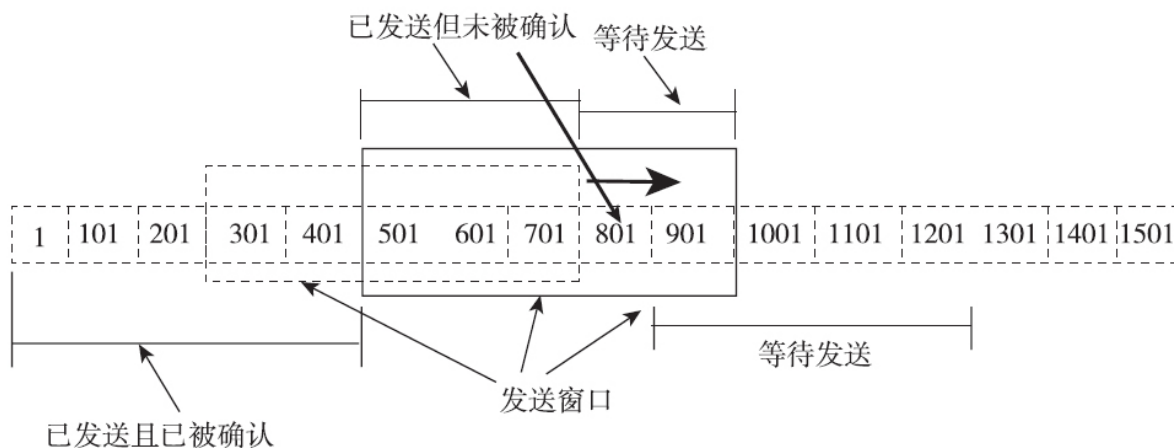


图 10-45 TCP滑动窗口示例

1) 首先假设现在发送端收到了接收端发来的一个确认数据段，“确认号”为301，“窗口大小”为500，表示可以连续发送5个数据段（起始序号为301）。左边那个虚线“发送窗口”代表的就是后面发送的301、401、501、601、701这五个数据段，此时因为已达到了对方窗口大小值，不能继续发送了，需要停下来等待对端的确认。

2) 如果某一时间收到了一个“确认号”为501的确认数据段（“ACK”字段值为1的数据段，下同），表示接收端已正确接收了301和401这两个数据段，即从“发送窗口”中删除这两个数据段，窗口向前滑动200字节（也就是301和401这两个数据段的大小），移到了图10-45中细实线“发送窗口”位置，其中501、601和701这三个数据段是原来已发送但没收到确认的缓存数据段，801和901这两个数据段才是要等待发送的。

假设以上返回的确认数据段中的“窗口大小”为400（这里起到了流量控制的作用），理论上说，发送端可以一次性连续发送4个数据段，但因为“发送窗口”中缓存了原来已发送的501、601和701这三个数据段（300字节），于是此次只能发送100字节，即801号数据段。

3) 如果此时收到了一个“确认号”为801的确认数据段，同时“窗口大小”字段值又为500了（这里也起到了流量控制的作用），则发送端知道接收端已收到了701号及以前所有的数据段了，于是从“发送窗口”中删除这些缓存的数据段，此时实际上缓存的数据段仅为801号数据段。但原来的901号数据段已在“发送窗口”中，等待发送，此时“发送窗口”只需要继续向前移300字节，也就是三个数据段（此时假设数据段大小均为100字节的情况下），即如图10-45所示的粗实线表示“发送窗口”，继续发送901、1001、1101、1201这四个数据段。

2.存在数据丢失情况下的流量控制

如果在数据传输过程中有一个或多个数据段丢失，则发送端接收不到对这些数据段的确认数据段，这时可以通过上节介绍的超时重传来解决。现在要探讨的一个问题是，如果某个时间，对端发送的数据段显示“窗口大小”字段值为0，这时发送端自然不能再发送数据了，只好等待对方发来一个“窗口大小”字段值不为0的数据段。可是如果对端发来的这个数据段在传输过程中丢失了，那么这时对于发送端来说，一直在等待来自接收端“窗口大小”字段值非0的数据段，而接收端又在

等待发送端发来新的数据，因为它自己不知道所发送的数据段在途中丢失了（当然这仅适用于C/S模式，对于非C/S模式，也就没有发送端和接收端之分了，双方都可以发送，都可以使用超时重传机制）。

就像你与你的朋友约定，在条件满足时他通过快递方式给你寄一样东西，但你并不知道他具体什么时候寄，他寄的时候也没告诉你（因为他只相信快递公司）。而恰好这件快递在途中丢失了，结果是你一直在等待你的朋友寄来你所需要的东西（你也不好意催问你的朋友是否寄了），而你的朋友一直在等待你收到的通知（总认为快递一般不会丢东西的）。这样可能等了好久，你的朋友才可能想起这件事，询问你是否收到了这件快递。

为了解决这一问题，TCP中引入了一个称为“持续计时器”（persistence timer）的定时器。在TCP连接的一端收到对端的一个“窗口大小”字段值为0的数据段时，启动该定时器。在这个定时器到期后，收到这个“窗口大小”字段值为0的数据段的一端会向对端发送一个非常小的探测数据段（一般仅携带1字节的数据），这时，对端在收到这个探测数据段后会返回一个确认数据段。如果在确认字段中的“窗口大小”字段值仍为0，则发送端重启上面的“持续计时器”，否则结合确认数据段中的“窗口大小”字段值和当前可用“发送窗口”大小，发送相应字节的数据，打破以上这种双方持续等待的局面。

10.5.2 基于传输效率的考虑

我们在前面已提到过，**TCP**数据段中只对“数据”部分的字节进行编号，原因是在对方传输层解封装后得到，并且所需要的只是“数据”部分内容，这样就可以很容易地知道接收的数据是否连续。否则，如果加上**TCP**数据报头，以及传输到网络层的**IP**报头，传输到数据链路层报头，数据部分的编号就不可能连续，况且不同数据段的报头大小都可能不一样，因为还有“可选项”部分。

对于大型的数据传输，这些报头加起来也才几十字节，对于一个数据段动辄上千个字节来说，这些协议头部分显然是可以忽略不计的，也就是说，传输效率是非常高的。但是我们要考虑各方面的应用，如在一些交互式应用中，每次传输的数据部分可能仅一个或几个字节，如果为每个这样的数据传输一次，显然传输效率是很低的，因为使用几十个协议头而最终传输的有用数据却仅几个字节。这类情况还会在发送确认数据段时经常发生，如果仅是用来确认的数据段，里面的数据量是非常小的，也正因如此，所以通常是在捎带大量数据的数据段中把**ACK**字段置1，同时起到确认的作用。

这时就涉及什么时候把这些数据组装成一个数据段进行传输的问题了。其中也涉及两个非常重要的算法：**Nagle**算法和**Clark**算法。

Nagle算法规定：如果数据每次以1字节的方式进入到发送端，则发送端只是发送第一个字节，然后其余的字节先缓存起来，直到发送出去的那个字节被确认为止。然后，将所有缓存起来的数据放在一个**TCP**数据段中发送出去，随后继续开始缓存后续字节，直到全部发送出去的字节全部确认为止。利用这种算法，可以大大提高传输效率，因为不会因1个字节而使用几十个协议头来携带。试想一下，例如，一个用户的键盘输入很快，而网络很慢，这时就会把用户的许多次输入字符组装成一个数据段，然后一次发送出去，大大减少所占用的网络带宽。**Nagle**算法还规定，当到达的数据已达到发送窗口大小的一半或者已达到数据段的最大长度时，也要发送出去。

还有一种情况，发送端的**TCP**实体从应用层接收到大量的数据，等待发送，而此时接收端的应用却每次只需要1字节的数据，然后向发送端发出确认数据段，当然，其中显示的“窗口大小”字段值就只是1了（1字节）。发送端收到确认数据段后自然也就只能再发送1字节的数据（但其中的**TCP**和**IP**协议头至少是40字节），然后接收端再次发回针对这1字节数据的确认数据段。显然，这种传输方式的效率也是很低的。

为了解决这种问题，**Clark**提出了一种解决方案，就是禁止接收端发送“窗口大小”字段值为1（也就是数据部分仅1字节）的数据段，即让接收端继续等待一段时间，使得接收端的“接收窗口”有足够的空间

可以容纳一个最大数据段长度的数据，或者它的缓存空间一半已空时（取两者的最小值）才发送确认数据段。这时，“窗口大小”字段值肯定不是1了，这样发送端就可以一次性发送更多的数据，从而提高传输效率。

10.6 TCP的拥塞控制

TCP中最复杂的功能可能就是拥塞控制功能了，因为两端经过的网络数量、网络类型、网络性能非常不确定，而这些因素又深深地影响着TCP连接拥塞控制的复杂性和难度。TCP的拥塞控制主要是依靠TCP连接双方商定的协议来减少数据的发送而实现的。

10.6.1 TCP拥塞控制简介

顾名思义，拥塞控制（congestion control）就是要控制“网络拥塞”的出现。什么情况下会出现网络拥塞呢？简单地说，网络中各个部分输入的流量大于输出的流量时就会出现网络拥塞。打一个比较类似的比喻（之所以说是“类似”，因为它更像“流量控制”类的比喻），就像在公路上发生塞车的一个根本原因就是出口道路太窄，而入口道路又比较宽，导致同一时间驶入的车辆数大于驶出的车辆数，最终使得这条路上排队并挤满了各种车辆，车辆的行驶速度自然会降下来。最坏的情况是什么，那就是完全“堵死”，车辆根本出不去，也进不来。这就类似网络中通常所说的“死锁”现象。

当然，道路上发生塞车的原因不仅只有上面所说的那一个，还有像上游驶入的车辆太多、中间或下游正在进行车辆检查、中间某些车

辆行驶速度太慢，或者发生了交通事故而占用了部分道路等。就像道路发生塞车的原因有多种一样，网络上发生拥塞的原因也可能是多方面的，而且网络结构越复杂，发生拥塞的原因也可能越复杂。如TCP连接的整个链路中有结点设备的缓存空间太小、数据转发能力太低、某段链路带宽太小、对端数据接收能力低等，都可能引起网络拥塞。而且往往是多个因素同时存在的，因此，处理拥塞控制问题不能简单地针对某一方面来加以解决，必须从全局角度来寻找解决方案，否则可能不仅不能解决拥塞现象，反而会形成新的瓶颈，使网络更加拥塞。

例如，用户想单方面地提高中间路由器结点的数据转发能力，而忽视了所经路径上各段链路的带宽，结果是虽然提高了路由器转发性能，也提高了数据转发速度，但也同时造成了在链路上排队前行的数据不断增多，这样不仅没有解决网络拥塞问题，反而使网络拥塞得更严重。再如，用户想单方面地提高路由器的缓存能力，但没有同步考虑路由器的数据转发能力和链路的带宽，结果是虽然可以使更多的数据在路由器上暂时缓存，但在缓存中排队的数据所需等待的时间会更长，因为队列比以前更长了，结果会因为超时重传这些数据。重传数据越多，网络负荷超越重，最终导致拥塞更加严重。就像我们在车站买票，如果售票员的售票速度没有提高，仅增加了用于排除等候的空间，就会让队列排得更长，这样是不能解决长时间排队买票难题的，反而使排在后面的人买到票的时间更长，同时使得售票厅更加拥挤。

如果把整个网络有效处理负荷的能力称为“吞吐量”（throughout），而把网络中发送端输入的负荷称为“输入负荷”（input load），则理想情况下它们之间有如图10-46所示的线性关系（吞吐量是呈45°斜线上升的）。

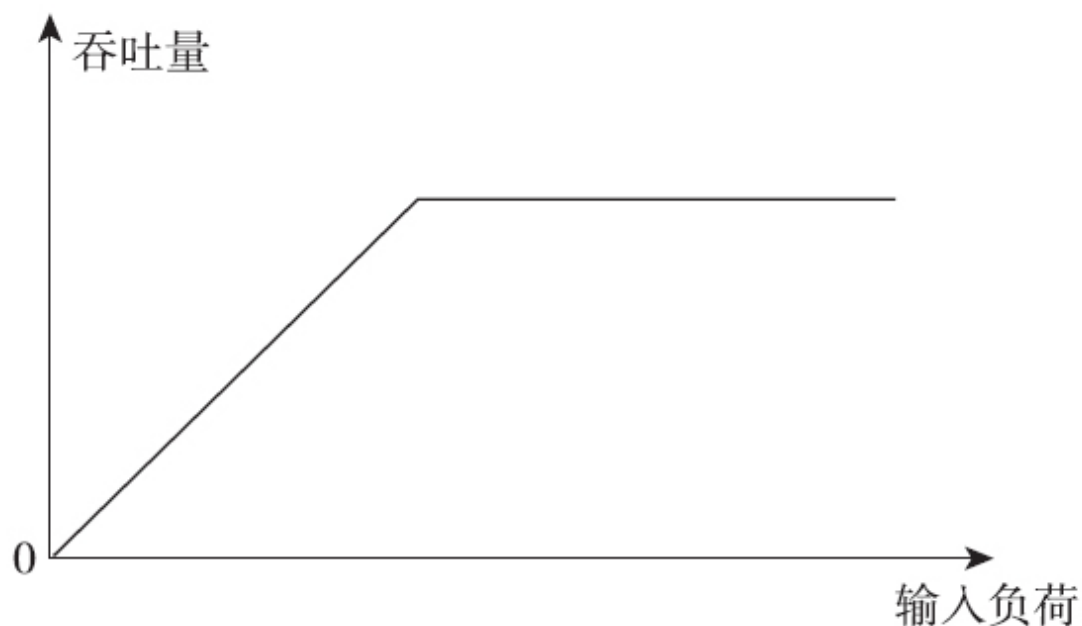


图 10-46 理想情况下的吞吐量与输入负荷之间的关系

从图10-46中可以看出，在开始时，网络系统的吞吐量随着输入负荷的增加呈同步提高态势（两者的提高量是一样的），但当到了一定时期（在输入负荷达到了网络的最大吞吐量时），输入负载再怎么增加，网络系统的吞吐量也不再提高了，而是保持在一个不变的水平。这时就出现了轻度拥塞现象。就像一条河流最初是没有水的，当上游河流放水时，它的水速会随着上游流入水的水速提高而提高，但当这条河流满了以后，也就是到了它的最大负荷水平时，它的水速不会再

有提高了，尽管上游流入的水速仍在提高。从上游新增的那部分水哪里去了呢？肯定是溢出了，就像网络达到最大吞吐量时，从发送端发来的超出吞吐量那部分的数据会被丢弃一样。

图10-46只是一种理想情况，实际上网络系统吞吐量与输入负荷之间的关系永远不会是线性关系。因为网络中不可能完全是理想状态，所以一开始“吞吐量”和“输入负荷”之间的关系就不是呈线性关系的。当输入负荷达到或接近最大吞吐量时，如果继续提高输入负荷，此时网络系统的实际吞吐量是不会保持在最大吞吐量水平的，而是呈现下降趋势的；如果此时输入负荷继续增加，那么最终可能导致网络系统的吞吐量下降到0，出现完全死锁状态，如图10-47所示。

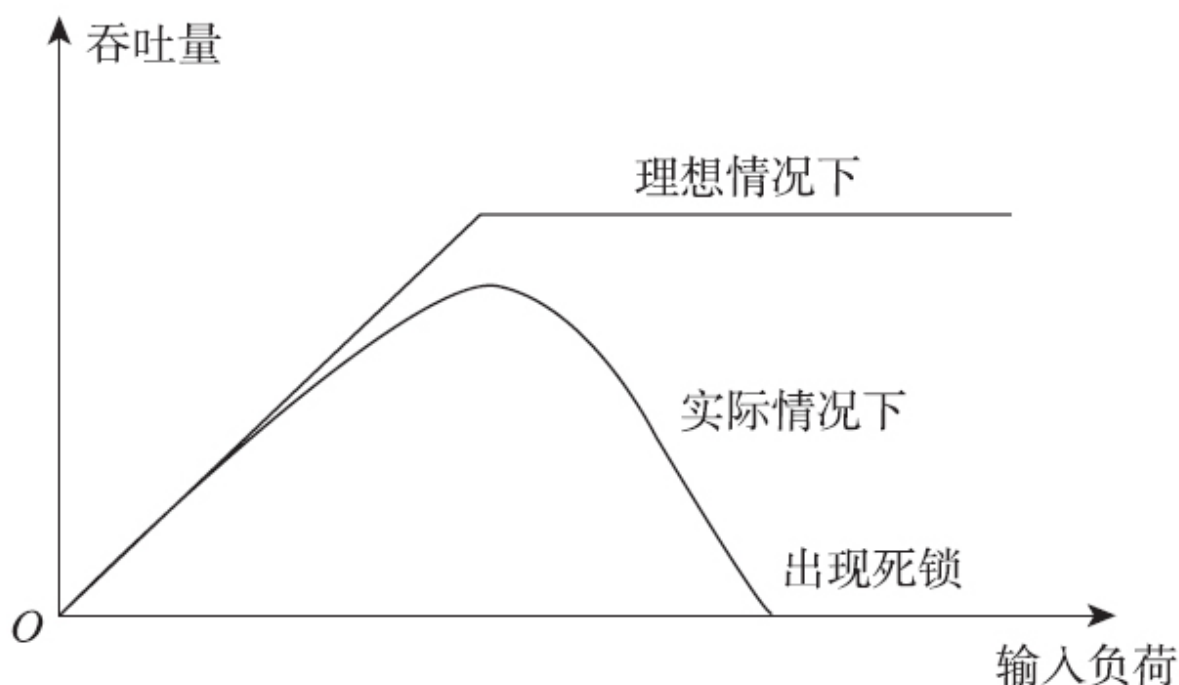


图 10-47 实际情况下的吞吐量与输入负荷之间的关系

10.6.2 TCP拥塞控制方案

虽然已经了解了发生拥塞的原因，也知道了网络系统吞吐量与输入负荷之间的关系，但要真正设计一套有效的拥塞控制方案却不是一件容易的事，因为拥塞的出现和所引起的因素都是动态的，是在不断发生变化的。发生网络拥塞的一个最明显的征兆就是出现了数据的丢失。于是，为了防止网络出现拥塞现象，出现了一系列的TCP拥塞控制机制。最初是由V.Jacobson在1988年的论文中提出的TCP的拥塞控制，由“慢启动”（slow start）和“拥塞避免”（congestion avoidance）组成，后来在TCP Reno版本中又针对性地加入了“快速重传”（fast retransmit）、“快速恢复”（fast recovery）算法。为了方便介绍，现假设数据是单方面发送的，另一个方向只传送确认数据段，而且假设接收端的窗口足够大，发送端的窗口大小由网络拥塞程度决定。

1.慢启动

慢启动是指为了避免出现网络拥塞而采取的一种TCP拥塞初期预防方案。其基本思想就是在TCP连接正式传输数据时，每次可发送的数据大小（这就是“拥塞窗口”的含义）是逐渐增大的，也就是先发送一些小字节数的试探性数据，在收到这些数据段的确认后，再慢慢增加发送的数据量，直到达到了某个原先设定的极限值（也就是下面将要提到的“慢启动阈值”（SSTHRESH））为止。

在“慢启动”拥塞解决方案中，发送端除了要维护正常情况下根据接收端发来的“窗口大小”字段值而调整的“发送窗口”外，还要维护一个“拥塞窗口”（Congestion Window, CWND），它是为了避免发生拥塞而设置的窗口，最终允许发送的字节数是这两个窗口中的最小值。如果从接收端返回的“窗口大小”字段值是100，而当时设置的“拥塞窗口”大小为50，此时只能发送50字节的数据。相反，如果从接收端返回的“窗口大小”字段值是100，而当时设置的“拥塞窗口”大小为200，则此时只能发送100字节的数据。下面仅假设“拥塞窗口”总是小于从接收端返回的“窗口大小”字段值（也就是“发送窗口”大小）。具体步骤如下：

- 1) 在一个TCP传输连接建立时，发送端将“拥塞窗口”初始化为该连接上当前使用的最大数据段大小（Maximum Segment Size, MSS），即CWND=MSS。然后它发送一个大小为MSS的数据段。

- 2) 如果在定时器过期前发送端收到了该数据段的确认，则发送端将“拥塞窗口”大小再增加一个MSS，也就是此时CWND大小为2MSS。然后发送2MSS大小的数据。

- 3) 如果这次发送的2MSS数据段也都被确认了，则“拥塞窗口”大小再增加两个MSS（此时相当于达到了4MSS），依此类推。

图10-48是一个采用“慢启动”方案的示例，其中M代表MSS，表示“最大数据段大小”。第一次发送一个MSS（即M1），收到M1的确认

后，再连续发送两个MSS（即M2和M3），当收到M3的确认（因为后面的已经确认的话，预示前面的都已正确接收，下同）后再发送四个MSS（即M4~M7），当收到M7的确认时，后面将发送8个MSS（即M8~M15）

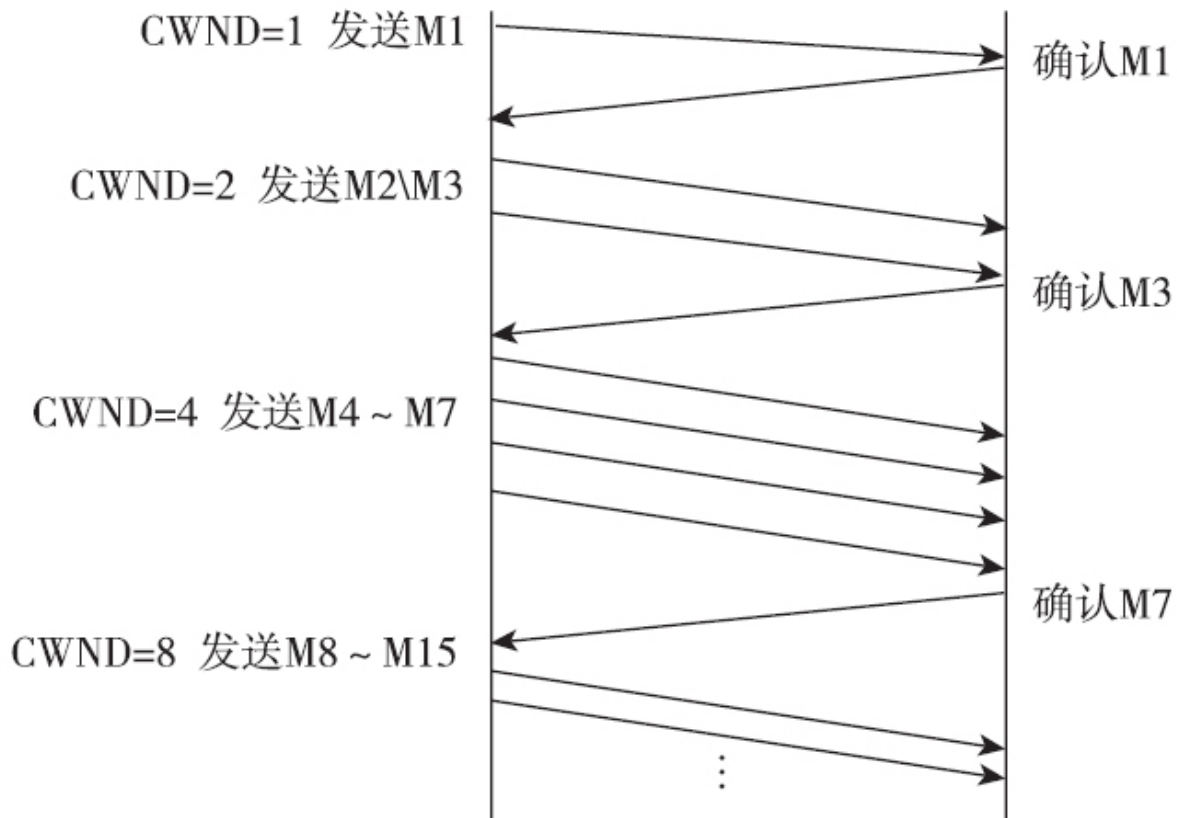


图 10-48 “慢启动”解决方案TCP数据发送示例

“慢启动”方案的基本规律：当“拥塞窗口”大小达到了 n 个MSS时，如果所有 n 个数据段都被及时确认，则新的“拥塞窗口”大小再增加 n 个MSS，也就是新的“拥塞窗口”大小是旧的“拥塞窗口”大小的2倍关系，

即“拥塞窗口”大小与最大数据段大小（MSS）的关系是2的指数关系，即1MSS、2MSS、4MSS、8MSS、16MSS、32MSS……

但是“拥塞窗口”不可能无限制地继续增大，即使CWND值仍小于从接收端返回的“窗口大小”字段值，都有可能在某个时刻出现数据丢失的现象。这个“拥塞窗口”大小是一个临界点，于是引入了“慢启动”方案的另一个重要参数——“慢启动阈值”（Slow Start Threshold, SSTHRESH），其初始值为64KB，即65535字节。当发生一次数据丢失时，SSTHRESH设为当前CWND的一半，而CWND又重新置为1MSS。然后继续使用“慢启动”方案来解决网络拥塞问题，不过当CWND再次增长到SSTHRESH（此时仅为原来CWND的一半）时便停止使用“慢启动”方案，需采用下面将要介绍的“拥塞避免”解决方案。

2.拥塞避免

当CWND再次大于或等于SSTHRESH时，启动“拥塞避免”解决方案。它的基本思想是在CWND值第二次达到SSTHRESH时，让“拥塞窗口”大小每经过一个RTT（一个数据段往返接收端和发送端所需的时间）时间仅值加1（即新的CWND只增加一个MSS大小，而不是原来CWND的几倍），使其以线性方式慢慢地增大，而不是继续像“慢启动”方案中那样以指数方式快速增大。显然，这种CWND增长速度明显要慢于“慢启动”方案中的CWND增长速度。当再次发生数据丢失时，

又会把SSTHRESH减为当前CWND的一半，同时把CWND置为1，重新进入“慢启动”数据发送过程，依此类推。

图10-49是一个“慢启动”和“拥塞避免”两种拥塞控制方案的控制示例。假设最初在某个时间的“拥塞窗口”大小（CWND）为64KB，而就在此后发生了数据丢失，没有收到后面发送的数据的确认，于是

（SSTHRESH）减为CWND的一半，即32KB（即图10-49中的“第一次调整的阈值”），同时把新的CWND设为1MSS，从坐标原点开始重新采用“慢启动”方案进行数据发送，当达到了SSTHRESH值（即32KB）时，采用“拥塞避免”方案进行数据发送，每个RTT时间CWND只增加一个MSS。但在第11次传输过程中（此时CWND=38KB）又发生了一次数据丢失，于是SSTHRESH再次降为当前CWND（即38KB）的一半

（即19KB），并且重新开始采用“慢启动”方案发送数据，直到达到新的SSTHRESH值（19KB），然后从这点开始又采用“拥塞避免”方案发送数据，依此类推。

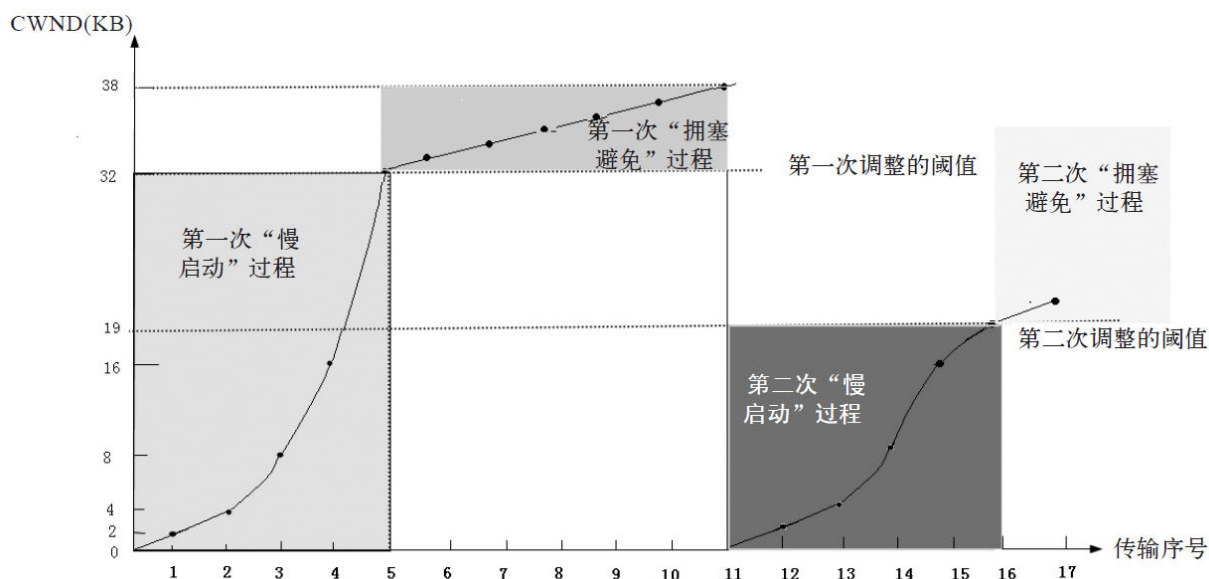


图 10-49 “慢启动”和“拥塞避免”方案数据发送示例

3.快速重传/快速恢复

上面介绍的“慢启动”和“拥塞避免”是1988年提出的拥塞控制方案，而1990年又新增了两种拥塞控制方案，就是下面要介绍的“快速重传”和“快速恢复”拥塞控制方案。

“快速重传”方案的基本思想：当接收端收到一个不是按序到达的数据段时，TCP实体迅速发送一个重复ACK数据段，而不用等到有数据需要发送时顺带发出确认；在重复收到三个重复ACK数据段后，即认为对应“确认号”字段的数据段已经丢失，TCP不等重传定时器超时就重传看来已经丢失的数据段。

为了进一步理解“快速重传”原理，现举一个如图10-50所示的示例。假设每次只发送一个大小等于MSS的数据段，并假设在第一次、

第二次发送M1、M2后，发送端都从接收端收到了确认，但第三次发送的M3在途中丢失了，而后面第四次发送的M4又收到了。正常情况，接收端是不会再发送任何确认数据段的，直到收到M3为止。然而，发送端此时并不知道M3丢失了，继续发送M4，在接收端收到M4时，为了尽快通知发送端M3还没有收到，于是再次发送一个M2确认数据段（其中的“确认号”字段值是M3的序号），相当于再次提醒发送端M3数据段没收到。但此时发送端还不会重发M3，希望再等等看，仍然继续依次发送M5、M6，在接收端每收到一个数据段后都会有一个重复的M2确认。在发送端收到了四个针对M2数据段的确认（一个确认M2，三个重复确认M2）时，发送端就会立即发送M3，尽管此时可能M3的重传定时器还没过期。

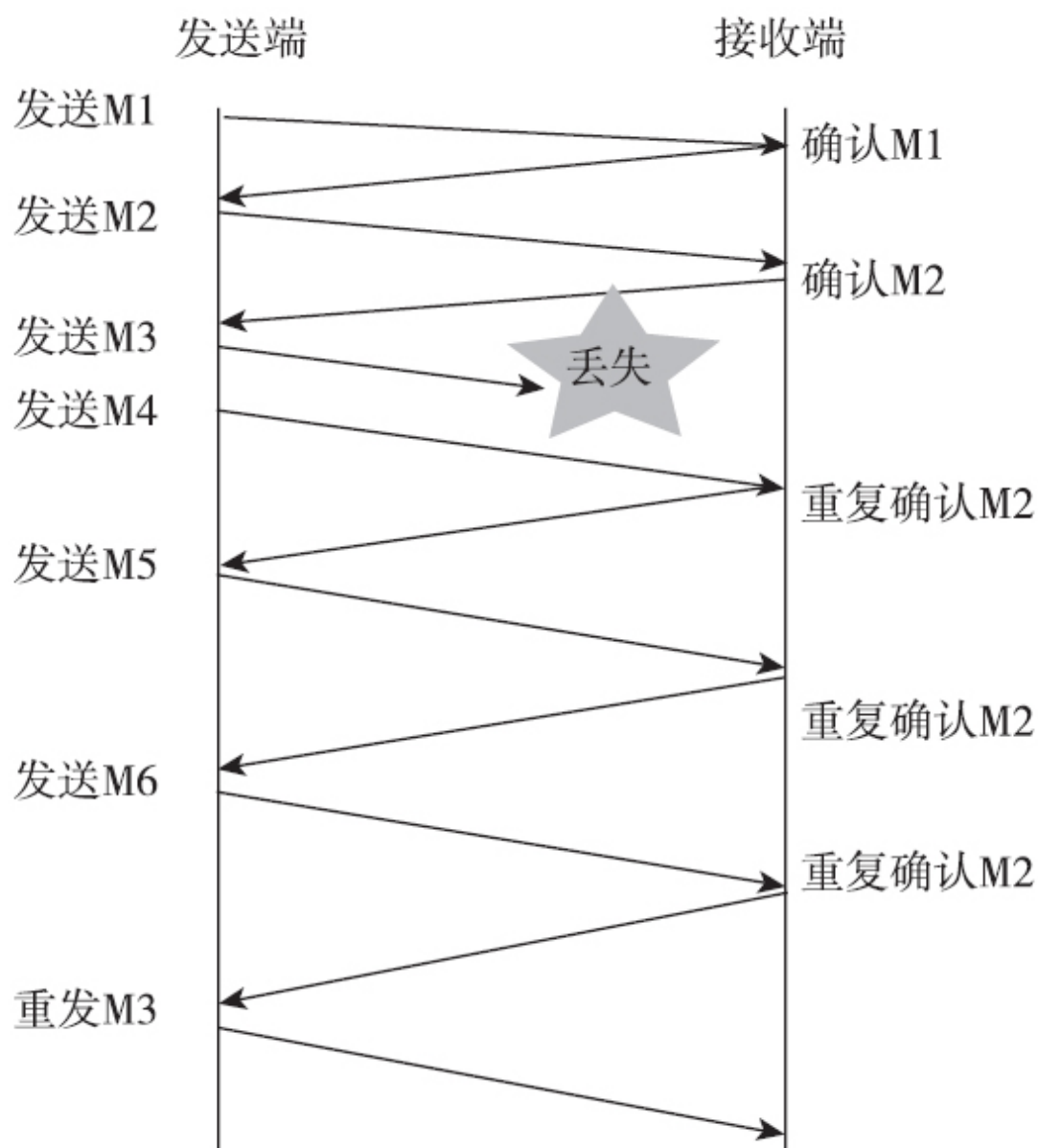


图 10-50 “快速重传”方案数据重传示例

在“快速重传”算法发送了看似已经丢失的数据段后，“快速恢复”算法同时开始发挥作用。“快速恢复”算法的基本思想：在收到第三个重复ACK时，把当前CWND值设为当前SSTHRESH值的一半，以减轻网

络负荷，然后执行前面介绍的“拥塞避免”算法，使CWND值慢慢增大，以避免再次出现网络拥塞。

10.7 UDP概述

在网络体系结构的“传输层”中，主要的传输层协议就是前面介绍的TCP，它是可靠的传输层协议，可以提供可靠的数据传输。然而，还有一些网络应用，应用的连续性和可靠性可能更重要，如视频会议、语音电话，其中丢失一部分数据的影响并不大，用户通过上、下部分的内容可能猜得出来，甚至这部分影响很难看得出来，但如果出现传输中断，或者延迟严重，可能就无法接受了。于是，出现了一种不面向连接的传输层——UDP（User Datagram Protocol，用户数据报文协议）。

10.7.1 UDP的基础知识

UDP是一种无连接传输层协议，不像TCP那样需要服务器监听，也不必等待客户端与服务器建立连接后才能通信，当然，最后能否把数据传输成功，UDP是不能保证的。

前面介绍的TCP采用的是面向连接的分组报文传输方式，而UDP采用的是无连接的数据报传输方式。分组报文头部是有明确的源地址和目的地址的，而数据报头部是没有这些信息的。该服务对消息中传输的数据提供不可靠的、尽最大努力的传送。这意味着它不保证数据

报的到达，也不保证所传送数据包的顺序是否正确。总体来说，UDP具有以下几个方面的明显特性：

(1) 无连接性

UDP可以提供无连接的数据报服务，这也决定了在使用UDP进行数据传输前是不需要建立专门的传输连接的，当然，在数据发送结束时也无须释放连接了。

(2) 不可靠性

因为UDP传输数据时是不需要事先建立专门的传输连接的，所以它的传输是不可靠的（但会尽最大努力进行交付），很可能传输不成功。UDP特别适用于一些短消息类的数据传输，如DHCP、DNS中的一些消息就是采用UDP进行传输的。

(3) 以报文为边界

UDP直接对应用层提交的报文进行封装、传输，但不拆分，也不合并，保留原来报文的边界。因此，UDP是报文流，而TCP是字节流。因为UDP不拆分报文，自然也就没有报文段之说，但UDP报文传输到网络层后，在网络层仍然可以根据网络的MTU值进行分割。

(4) 无流量控制和拥塞控制功能

使用**UDP**进行数据传输时不能进行流量控制和拥塞控制，因为这类数据传输的连续性要比数据的完整性更重要，允许数据在传输过程中有部分丢失，如**IP**电话、流媒体通信等。

(5) 支持各种交互通信方式

TCP不支持组播、广播通信方式，只支持一对一的单播方式，但**UDP**支持各种通信方式，即可以是一对一、一对多、多对一和多对多的方式。

计算机网络中有许多使用**UDP**的应用服务，如**DNS**、**SNMP**、**DHCP**和**RIP**等。

10.7.2 UDP数据报头部格式

UDP数据报格式比较简单，如图10-51所示，其中“数据”字段是来自应用层的报文内容，其他部分为UDP协议头，总共8字节。下面是UDP数据报头部各字段的说明：

16	16	16	16	
源端口	目的端口	长度	校验和	数据

图 10-51 UDP数据报格式

(1) 源端口

源端口字段用来标识源主机上使用的UDP端口，占16位（2字节）。这个字段是可选的，仅当需要目的主机返回一个应答时才有意义。如果不使用它，则此字段值为0。

(2) 目的端口

目的端口字段用来标识目的主机上使用的UDP端口，占16位（2字节）。如果目的主机应用层没有对应端口的应用进程，则该UDP数据报会被丢弃。

(3) 长度

长度字段用来标识此UDP数据报的长度（包括UDP数据报头部和“数据”部分），以字节为单位，占16位（2字节）。因为一个IP分组的最大长度为65535字节，所以，理论上的UDP数据报“数据”部分的最大长度为65535-20（最小IP报头）-8（UDP）=65507字节。但实际上在互联网中通常是限制在512字节以内，这主要是受互联网结构比较复杂、接入性能比较低影响的。

（4）校验和

在进行校验和计算时，需要在UDP数据报头部前加上12字节的伪头部（如图10-52所示，它不是数据中的真实成分，也不用来传输，仅用来计算校验和），然后对整个UDP头部和“数据”部分进行校验。通常是采用CRC校验方式，具体参见7.3.4节相关内容。

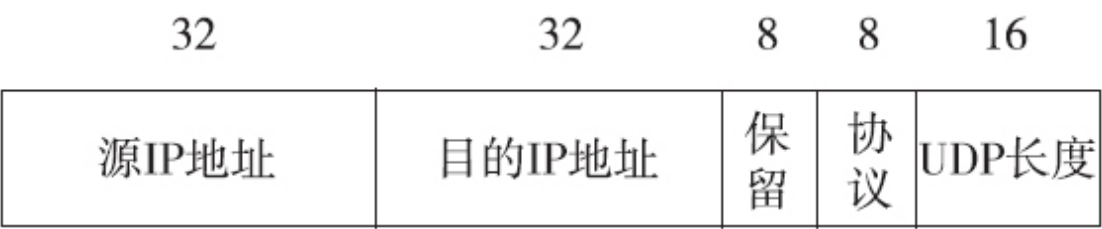


图 10-52 UDP伪头部

因为UDP无连接管理，无差错控制，也无拥塞控制，所以比较简单。

第11章 应用层

在OSI/RM体系结构的“传输层”之上、“应用层”之下有“会话层”和“表示层”，但实际上这两层的功能都非常简单，因此，在TCP/IP体系结构中，这两层的功能整合到“应用层”中，现在的各种网络应用程序也是按照这种结构来进行程序设计的。本书不再单独介绍“会话层”和“表示层”，而是直接介绍体系结构的最高层——应用层。

前面各章节介绍的内容主要完成两方面的功能，一是网络连接的建立（或者说是网络通信平台的搭建），二是数据传输通道（包括数据链路层构建的局域网内部点对点数据传输通道和传输层构建的不同网络间端对端数据传输通道）的建立。它们的最终目的只有一个，那就是为运行的网络应用搭建有效的平台，这就是本章将要介绍的内容，即OSI/RM和TCP/IP体系结构的最高层——“应用层”所提供的应用服务。但要注意的是，应用层提供的同样是服务，而不是提供具体的网络应用软件。但与其他各层一样，在网络体系结构中，每一层都有一个服务作用实体，“应用层”的实体（AE）就是各种正在运行的用户网络应用进程。

随着网络技术的发展和网络应用的普及，各方面的网络应用也不断涌现。每类网络应用都需要对应的应用服务支持，“应用层”要解决的问题就是为用户提供所需的应用服务。这些网络应用服务都是C/S

（客户端/服务器）工作模式，当然，有的是客户端与服务器分离（也就是一台主机要么担当客户端角色，要么担当服务器角色，不能同时安装两种角色的服务软件），有的是两者同时支持。本章主要介绍HTTP（Web服务）、E-mail（电子邮件服务）、DNS（域名解析服务）、DHCP（动态主机配置服务）四类网络应用服务的客户端和服务器的基本工作原理。

11.1 应用层概述

应用层是OSI/RM和TCP/IP体系结构的最高层，通过使用下面各层所提供的服务，直接向用户提供服务，是计算机网络与用户之间的界面或接口。就像其他各层所提供的服务一样，应用层的各种服务功能也是通过具体的通信协议来实现的。

应用层的网络应用服务非常多，而且有许多是用户每天在使用的，如Web服务、电子邮件服务、DNS（域名服务）、DHCP（IP地址自动分配服务）、文件传输服务、远程登录服务等。随着互联网应用的普及和发展，各种新的网络应用服务层出不穷，本章仅介绍在TCP/IP网络中最常用的一些服务。

11.1.1 应用层组件及典型应用服务

为了向用户提供有效的网络应用服务，应用层需要确立相互通信的应用程序或进程的有效性并提供同步，需要提供应用程序或进程所需要的信息交换和远程操作，需要建立错误恢复的机制以保证应用层数据的一致性。应用层为各种实际网络应用所提供的通信支持服务统称为ASE（Application Service Element，应用服务组件）。

不同的ASE可以方便地让各种实际的网络应用与下层进行通信。其中，最重要的3个ASE分别是ACSE（Association Control Service Element，关联控制服务组件）、ROSE（Remote Operation Service Element，远程操作服务组件）和RTSE（Reliable Transfer Service Element，可靠传输服务组件）。ACSE可以将通信两端用户所使用的应用程序名关联起来，用于在两端应用程序之间建立、维护和终止连接；ROSE采用类似远程过程调用的请求/应答机制实现远程操作；RTSE则通过先将数据转化为8位字节串，然后将数据串分解为多个段，最后将每段传送到表示层，以便发送在分段之间建立检验点。通过表示层服务，RTSE使用会话层活动管理服务来管理数据分段的传输。

除了以上提到的3个应用服务组件外，OSI/RM体系结构的应用层提供了5种不同的网络应用类型来实现不同的应用需求，分别是MHS（Message Handling System，报文处理系统）、FTAM（File Transfer, Access and Management，文件传输、存取和管理）、VTP（Virtual

Terminal Protocol, 虚拟终端协议)、DS (Directory Service, 目录服务)、TP (Transaction Processing, 事物处理)、RDA (Remote Database Access, 远程数据库访问)。但由于目前OSI/RM体系结构只是起到参考模型的作用, 所以并没有实际的网络应用是按照上述协议实现的。而在TCP/IP体系结构中的应用层却相反, 拥有许多主流的应用层协议和基于这些协议实现的TCP/IP应用。

TCP/IP体系结构中的应用层解决了TCP/IP网络应用存在的共性问题, 包括与网络应用相关的支撑协议和应用服务两大部分。其中的支撑协议包括域名服务系统 (DNS)、动态主机配置协议 (DHCP)、简单网络管理协议 (SNMP) 等; 典型的应用服务包括Web浏览服务、E-mail服务、文件传输访问服务、远程登录服务等, 另外, 还有一些与这些典型网络应用服务相关的协议, 包括超文本传输协议 (HTTP)、简单邮件传输协议 (SMTP)、文件传输协议 (FTP)、简单文件传输协议 (TFTP) 和远程登录 (Telnet) 等。

11.1.2 应用层的C/S服务模型

网络应用软件之间最常用、最重要的交互模型是C/S（Client/Server，客户端/服务器）模型。例如，Web服务、E-mail服务、DNS服务、DHCP服务、FTP服务等都是以这种模型为基础工作的。这里所说的“客户端”和“服务器”分别是指通信双方主机上安装的对应用应用程序。

网络上的应用程序之间为了能够顺利通信，一方通常需要处于守候状态，等待另一方请求的到来。这就是本节所介绍的C/S模型，首先服务器需要时刻守候各种应用服务（通常是通过监听各个传输端口来实现），然后客户端向服务器发出对应的应用会话请求，最后服务器对客户端的会话请求做出响应。如图11-1所示的是TCP/IP体系结构中一个通过Internet进行交互的C/S模型示例。在图11-1中，服务器处于守候状态，并监听客户端的会话请求，在客户端发出应用会话请求命令，并经Internet传输给服务器后，服务器接收到这个会话请求，并执行相应会话请求所指定的任务，最后将执行的结果以应答消息经Internet返回给客户端。

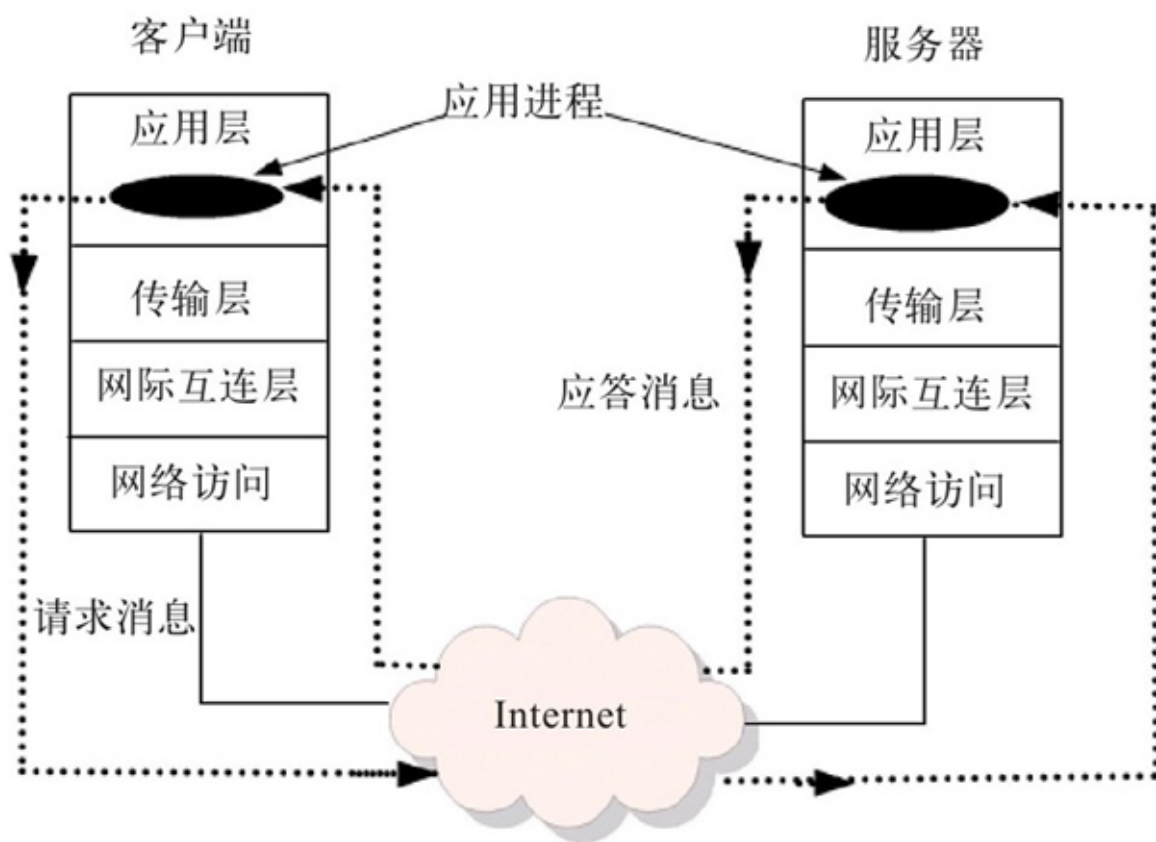


图 11-1 应用服务的C/S模型示例

一台主机上通常可以运行多个应用服务器程序，每个服务器程序需要并发地处理多个客户的请求，并将处理的结果返回给客户端。因此，服务器程序通常比较复杂，对主机的硬件资源（如CPU的处理速度、内存的大小等）及软件资源（如分时、多线程网络操作系统等）都有一定的要求。而客户端程序功能则相对简单，通常不需要特殊的硬件和高级的网络操作系统。

C/S模型不但很好地解决了互联网应用程序之间的同步问题（何时开始通信、何时发送信息、何时接收信息等），而且C/S模型的这种非

对等相互作用的特点很好地适应了互联网资源分配不均的客观事实，因此成为互联网应用程序相互作用的主要模型。

在TCP/IP互连网络中，服务器程序通常使用TCP或UDP的端口号作为自己的特定标志。在服务器程序启动时，首先在本地主机注册自己使用的TCP或UDP端口号，这样服务器程序在声明该端口号已被占用的同时，也通知本地主机如果在该端口上收到信息，则需要将这些信息转交给注册该端口的服务器程序处理。在客户端程序需要访问某个服务时，可以通过与服务器程序使用的TCP端口建立连接（或直接向服务器程序使用的UDP端口发送信息）来实现。

在互连网络中，客户发起请求完全是随机的，有可能出现多个请求同时到达服务器的情况。因此，服务器必须具备处理多个并发请求的能力，可以使用以下两种方案。

(1) 迭代服务器 (iterative server) 方案

在迭代服务器解决方案中，服务器程序中包含一个请求队列，客户请求到达后首先进入队列中等待，服务器按照先进先出的原则对这些客户端请求一个个做出响应，即服务器只有处理完前一个请求后才会处理下一个请求。

(2) 并发服务器 (concurrent server) 方案

并发服务器是一个守护进程，在没有请求到达时它处于等待状态。一旦客户端请求到达，服务器立即再为之创建一个子进程，然后回到等待状态，由子进程响应请求。当下一个子进程到达时，服务器再为之创建一个子进程。其中，并发服务器称做主服务器，子进程称做从服务器。

11.2 Web服务基础

目前网络应用中普及度最高、应用频次最多的是通常所说的万维网（World Wide Web，WWW）服务，或者称为Web服务。Web服务的核心应用层协议是HTTP（Hypertext Transfer Protocol，超文本传输协议），也可以把HTTP看成是Web服务的“幕后英雄”。

进行网站访问时好像觉得非常简单，点一下鼠标，或者输入一个网站名称就进去了，其实它仍需要一整套Web服务功能来实现，其中涉及许多协议和网络应用程序功能。下面先来分析一下Web服务模型（或者称为Web服务系统结构）。

11.2.1 Web服务模型

任何网络应用服务都不是单一程序或者单一命令可以实现的，它们必须有一个能实现服务请求、服务提供，甚至包括服务注册的完整系统结构，就是通常所说的“服务模型”。

从用户的角度来看，Web服务，或者所有互联网网站其实就是一个全球范围内的巨大文档，更具体地说就是Web页面集合。有了上述基本认识，可以很容易得出Web服务模型，如图11-2所示。Web服务模型包括3个部分：Web服务提供者（Web服务器）、Web服务请求者（Web客

户端)和Web服务注册中心(互联网注册、管理机构),其中“Web服务注册中心”是可选项,仅在互联网中的Web服务才是需要注册的,而局域网中是不需要的。

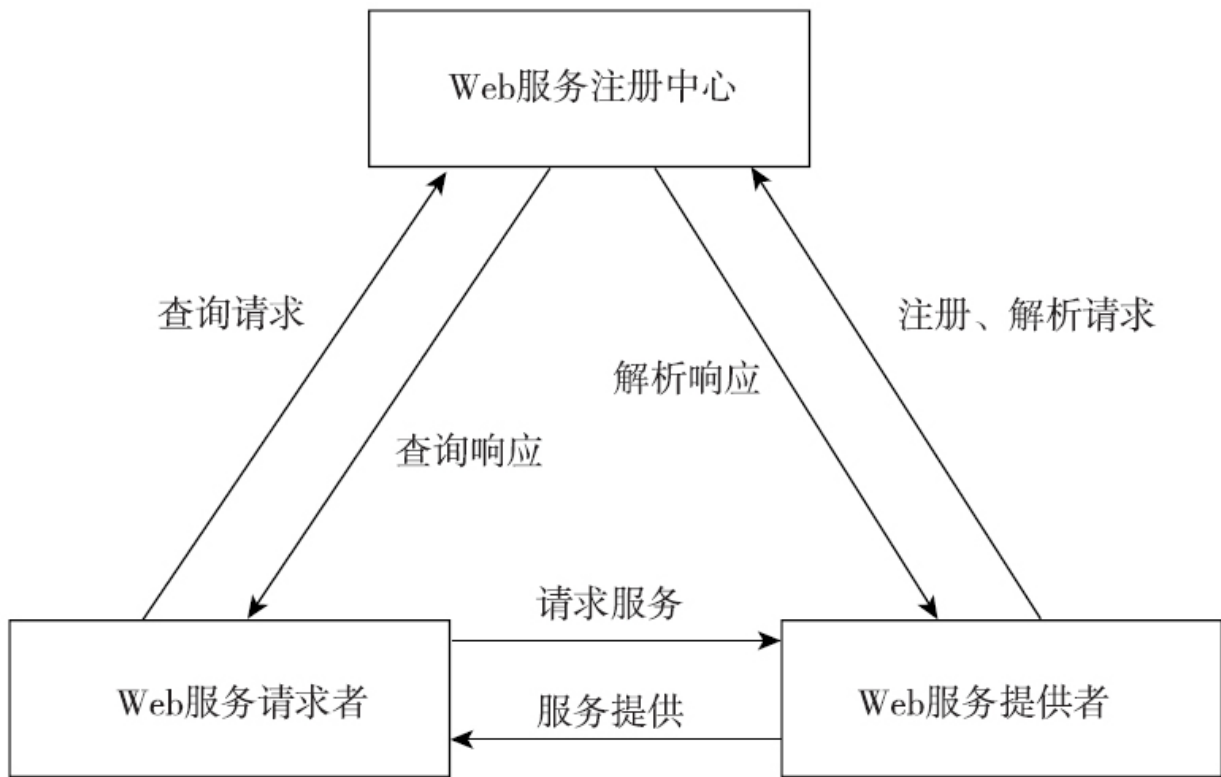


图 11-2 Web服务模型

下面简单介绍Web服务模型中3个组成部分的基本功能。

□Web服务注册中心：指互联网注册、管理中心和提供网站域名解析的ISP（Internet服务提供商），是一个可选部分，而在仅用于局域网内部的网站访问中无须使用此部分。它负责互联网网站的注册和管

理，并向用户提供互联网域名解析服务。因为它主要涉及域名解析服务——DNS，所以在本章后面介绍DNS服务时再进行详细介绍。

□Web服务提供者：指Web服务器，或者称做Web网站，是HTTP服务器端。它是由一些Web服务器程序（如IIS、Apache等）开发的，除了要对网站本身进行描述，并向注册中心注册外，更重要的职责是为Web访问用户提供所需的网页信息。

□Web服务请求者：指Web用户在访问网站时所用的HTTP客户端程序，如各种浏览器（如IE、Chrome浏览器等）。在Web浏览器中输入网站的域名或IP地址，或者在其他网站或文档上单击该网站的链接即可进入到对应的网站，然后再单击相应的页面链接访问所需要的文字、图片、音/视频等页面。

以上各组成部分均包括一些协议，并由支持这些协议的网络应用软件来实现。本节仅介绍“Web服务提供者”和“Web服务请求者”这两部分的功能实现原理。下面先了解一下万维网的一些基本特性。

11.2.2 万维网的全球统一标识

如果没有一个基于全球范围的唯一标识，那么要在互联网中找到你需要访问的那一个网站或网页，是无法想象的。在万维网中，存在3种全球统一标识，分别是URL（Uniform Resource Locator，统一资源定位器）、URI（Uniform Resource Identifier，统一资源标识符）和URN（Uniform Resource Name，统一资源名称）。

这3种标识不仅看起来非常相似，而且关系也非常密切。从它们的中文名称可以看出，URL是用来定位资源的，标识的是访问对应资源的路径，而URI是用来标识资源的，URN则是用来为资源命名的。注意，URI中的标识符可以是资源的名称，也可以是资源的地址。另外，URI在RFC1630中定义，发布于1994年6月；URL在RFC1738中定义，发布于1994年12月；URN在RFC1737中定义，发布于1994年12月。从中也可以看出，是先有URI，然后才有URL和URN，URL和URN是URI扩展后的子集。下面对它们分别进行介绍。

1.URL

互联网上的网站，无论它的服务器位于哪里，用户在连接上互联网的计算机的浏览器中输入这个网站的网址都可以轻易访问到该网站，或者该网站上的对应页面。这就是Web网站上所有资源的统一定

位标识特性——URL，它是URI的一种。URL标识一个互联网资源，并指定对其进行操作或取得该资源的方法。这里所说的“资源”是指在互联网上可以访问的任何对象，包括文件目录、文件、文档、图片、图像、音/视频，甚至是电子邮件地址等。

URL采用了全球统一的格式来标识万维网上资源的访问地址，以确保每个互联网网站和网站页面在全球范围内都有唯一的地址标识。URL的一般格式（其实后面还可以有可选项）为：

<协议>://<主机名>:<端口>/<路径>

（1）协议

这里的“协议”参数是指访问对象所使用的协议，包括HTTP、FTP、Gopher、Telnet、File、Mailto等。注意，在协议后须加上“://”。

（2）主机名

“主机名”参数是指Web服务器名称，互联网上的服务器主机名就是网站的域名，用来指定用户要访问的是哪台服务器主机。互联网的主机名通常是以www开始的（但不全是），后面必须跟着“:”（冒号），以指定访问对象建立通信连接时所用的传输层端口。

（3）端口

“端口”参数用来指定在建立连接时所用的传输层端口号。如果采用的是常规协议，端口号可以省略，直接使用默认的端口号，如HTTP的默认端口号为80，FTP的默认端口号为21，Telnet的默认端口号为23。即使是常规协议，所使用的端口号也可以更改，但一定要与Web服务器上设置的端口一致，否则访问不了。端口号后面跟的“/”代表服务器的根目录。

(4) 路径

“路径”参数用来指定要访问的对象在Web服务器上的文件路径，与本地主机上的文件路径格式一样，以根目录（/）开始。如果要访问的是网站首页，则不用输入此部分，直接以前面的“/”符号访问服务器的根目录即可。

如果要访问51CTO网站，则可以在浏览器地址栏中输入：
`http://www.51cto.com`，当然，现在的浏览器可以不用输入“`http://`”和“`/`”这两部分了，浏览器会自动为用户加上的，另外，没有指定端口时，直接使用HTTP的默认端口号80。

如果要直接访问在51CTO博客中的一篇文章，那么可以在浏览器地址栏中输入如下地址：`http://winda.blog.51cto.com/55153/868663`，其中，“`winda.blog.51cto.com`”是在51CTO博客上的博客域名，后面的“`/55153/868663`”是这篇文章的路径。

如果要访问一个域名为ftp.winclass.net的FTP站点，则可表示为ftp://ftp.winclass.net，当然，这里是没有考虑访问用户账户的；如果要访问一个电子邮箱winda@wincalss.net，则可输入：
mailto:winda@wincalss.net。

2.URI和URN

上文提起到过，URL和URN都是URI的子集，在应用上可视为URL、URN或两者兼备。URL上文介绍过，而URN仅是指资源的名称，如国际书号、互联网电子邮箱账户、RFC文档等。URN如同一个人的名称，而URL代表一个人的住址，但这两种统一标识实际上使用频率并不高，现在主要用的还是URL。

URI的基本语法格式如下：

<方案名> : <主机名> / <路径>

其中，“方案名”是指对资源进行标识的协议（但有时不是直接用协议名本身显示）。后面的“主机名”是指保存资源的服务器域名，而“路径”是资源的保存路径，它不会随访问路径变化而变化。下面是几种常见URI类型的基本格式。

□mailto:mbox@domain：指定通过SMTP发送电子邮件的电子邮箱地址。

□ftp://user:pass@server/file: 指定通过FTP传输文件的用户账户和文件位置。

□http://domain/path: 指定通过HTTP访问的网站或网页的路径。

□file:///path: 标识在本地磁盘或者网络文件系统上的文件地址。

其中，第一种类型既可以说是一个URN，也可以说是一个URL，本例中标识的是一个电子邮箱名称，而其他3个是URL。

11.2.3 万维网文档标记

因为Web用户使用的操作系统、浏览器类型多种多样，这些操作系统和浏览器所使用的开发语言也可能完全不同，所以为了在不同用户计算机上都能正确地显示万维网中的网页信息，也必须为网站的信息制定全球统一的表示方法（称为“标记”方法），那就是本节要重点介绍的HTML（HyperText Markup Language，超文本标记语言）。

HTML是一种万维网的标记语言（不是一种开发语言），用来结构化信息，如标题、段落和列表等，也可用来在一定程度上描述文档的外观和语义，由万维网协会（World Wide Web Consortium，W3C）维护。使用HTML开发的网页文件扩展名为.html（早期DOS时代采用的是3位扩展名，扩展名为.htm）。目前主要使用的是HTML 4.0版本，不过现在HTML 5.0版本已经发布，并即将全面进入实用阶段。

HTML使用一套标签（tag）来标记网页内容的格式并进行排版。HTML标记标签通常称为HTML标签（HTML tag）。HTML标签是由尖括号（< >）括起来的关键词（关键字不区分大小写，但通常以小写表示），比如<html>。而且HTML标签通常是成对出现的（但不是所有标签都是成对的，如换行标签
和分段标签<p>，它们只有一个标签），后面一个标签前面要加上“/”符号，比如和，第一个标签是开始标签，第二个标签是结束标签，开始标签和结束标签

也称为开放标签和闭合标签。例如，<i>表示后面的字体采用斜体排版，</i>表示此次采用斜体排版结束。常见的HTML标签如表11-1所示。更详细的HTML知识可参见相关资料。

表 11-1 常见 HTML 标签

标签	含义说明
标记网页结构类标签	
< html > < /html >	标签中间部分为网页内容
< head > < /head >	标签中间部分为网页头部
< title > < /title >	标签中间部分为标题内容
< body > < /body >	标签中间部分为网页正文
版面排版类标签	
< h# > < /h# >	定义标签中间部分内容标题字号
< font size=# > < /font >	定义标签中间部分内容字体大小
< font face=" 字型名称 " > < /font >	定义标签中间部分内容字型
< font color=#rrggbb > < /font >	定义标签中间部分内容文字颜色
< small > < /small >	定义标签中间部分内容为小字体
< big > < /big >	定义标签中间部分内容为大字体
< b > < /b >	定义标签中间部分内容为粗体字
< i > < /i >	定义标签中间部分内容为斜体字
< u > < /u >	定义标签中间部分内容为下划线字
< strike > < /strike >	定义标签中间部分内容为删除线字
< sub > < /sub >	定义标签中间部分内容为下标字
< sup> < /sup >	定义标签中间部分内容为上标字
< br >	从此处换行
< p >	从此处分段

11.2.4 HTML文档类型

上面讨论的HTML标记类文档在磁盘中保存时是以.html作为文件扩展名的。保存后，文档的内容是不会改变的，Web每次打开这个文档所见到的内容都是完全一样的，除非经过了人为编辑、修改。这就是通常所说的“静态HTML文档”的概念。

在静态HTML文档中，网页内容是静态分配的，用户每次访问时所看到的内容是一样的。但这里要区分两个概念，那就是“静态HTML文档”与“静态HTML元素”，这两者不是等同的概念，也没有一一对应的关系。也就是说，在静态HTML文档中可以有像文本、图片之类的静态网页元素，另外，HTML 4.0及更新版本还可标记图像、动画、音/视频这类动态网页元素。

与静态HTML文档相对的是动态HTML文档。动态HTML文档不是由管理员事先制作好的网页，而是即时由服务器应用程序或者脚本程序，根据当前应用和用户提交的表单数据自动生成的，其显示的内容是在不断变化的。例如，我们在网络购物时，每次在提交订单时都会自动生成一个页面，其中显示了我们当次购物的商品、金额及收货地址信息等。尽管整个网页结构每次都是一样的，但网页中的内容也许

每次都不一样，比如购物商品、金额等信息。这就是动态HTML文档。

既然动态HTML文档不是事先制作并保存在服务器磁盘上的，就需要有专门的网络应用程序来生成。当用户在网站上单击选择了某类应用时，Web服务器就会调用对应的应用程序，然后把控制权交给该应用程序。而该应用程序接收用户在表单中输入的数据，用户确认提交表单后应用程序就会对所提交的数据根据对应的网络应用程序功能进行处理，然后在用户浏览器中输出处理的结果。由于用户每次提交的数据，或者应用环境不一样，所以可能每次最终从Web服务器端返回的页面内容都是不一样的，就像我们每次在网站上购物所得的结算单都不一样。例如，在铁道部网站查询车票时，不同时间得到的结果也是不一样的，因为每时每刻可售票数都在变化。

通过上面的分析可以看出，动态HTML文档的创建至少要满足两个条件：第一个条件是在Web服务器端要有能自动生成动态HTML文档的网络应用程序。这个读者可以不用去了解，因为在进行Web服务器开发时，程序设计师会做这方面的工作。第二个条件是Web服务器要能将用户在表单中输入的数据发送到对应的网络应用程序，然后Web服务器又能正确地把应用程序处理的结果以HTML文档方式回显给用户。这就是一种逻辑接口——CGI（Common Gateway Interface，公共网关接口）的功能。

CGI是一种基于浏览器的输入，并可在Web服务器上运行应用程序（统称CGI程序）的方法。它允许Web服务器与后端程序及脚本进行通信，同时后端程序和脚本又可接收用户的输入信息（主要是来自表单），最后还能以HTML页面作为用户进行表单提交后的响应，如图11-3所示，具体描述如下（注意其中的序号与步骤是对应的）。

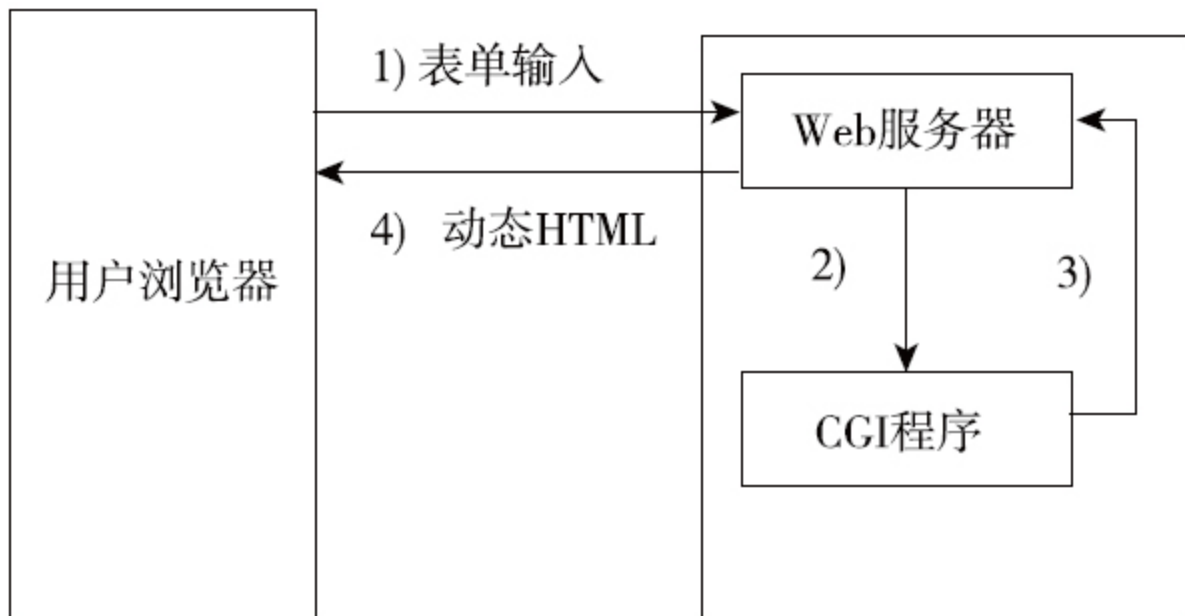


图 11-3 利用CGI程序生成动态HTML文档的基本流程

1) 用户在网站上单击了某项应用（如单击购买一个商品）后，相当于向Web服务器提交了一个表单。

2) Web服务器在收到用户提交的表单后，会通过服务器程序中已设置好的关联功能，把这个表单传送给对应的应用程序（也就是CGI程序）。

3) CGI程序接收用户提交的表单，并对其进行处理（其中可能需要调用其他服务器，或者客户主机，如各种数据库服务器），然后通过CGI程序的动态HTML文档生成功能，把处理结果以一个自动生成的HTML返回到Web服务器。

4) Web服务器再把这个自动生成的HTML回显到Web用户的浏览器上。

CGI程序其实都是脚本（script）类程序。“脚本”通常是一些可执行命令（往往带有许多参数）的集合，但需要调用其他程序来解释或执行，不是计算机可直接处理的。这类解释或执行脚本的程序就是各种编程语言，特别是一些专门的脚本语言，如Perl、JavaScript和VBScript等。

但是，CGI程序并不是生成动态HTML文档的唯一方法，还有一种目前广被采用的做法——在HTML页面中嵌入一些由编程语言编写的脚本（许多编程语言都可编写脚本，如PHP、JSP、ASP等，它们都可以接收并处理表单输入，同时与Web服务器进行交互），然后让Web服务器自己来执行这些脚本，以便生成最终的动态HTML文档并发送给客户。这样一来，也就不需要Web服务器调用CGI类应用程序了，直接由Web服务器来完成，效率更高。当然，这仅适用于一些小的消息类动态HTML页面。

11.2.5 HTML文档的“三超属性”

万维网中有3个“超”字非常具有吸引力，它们分别代表“超文本”、“超媒体”和“超链接”。我们姑且把它们称为万维网中的“三超属性”。

“超文本”（HyperText）其实也是一种文本格式，只是它可用下面将要介绍的“超链接”的方法，将保存在不同位置的文档或者文本信息组织在一起，构成一个相互关联的网状文本。例如，在访问网站时，经常见到有一些文字呈不同颜色（一般为蓝色），用鼠标单击它们后便弹出另外一个网页，或者消息框，用于解释那部分制作链接的文字。

“超媒体”（HyperMedia）其实可以看成是前面“超文本”属性的扩展，把“超文本”中的文本媒体保存格式扩展到其他所有格式。早期的网站基本上是一些静态甚至纯文本内容，现在的网站的内容就丰富多了，除了传统的文本信息、静态图片外，还有动态图像、动画、音/视频等各种媒体格式。而且，这么多种格式都可以在同一个网站互存，并可以相互链接。但目前通常是把“超媒体”等同于“超文本”，因此现在的超文本标记语言中不再只可以标记文本格式，还可以标记如图片、图像、动画、音/视频等各种媒体格式。

“超链接”（HyperLink）是HTML文档的最重要属性，是指从一个网页指向一个目标的连接关系。这个目标可以是另一个网页，也可以是相同网页上的不同位置，还可以是一个图片、一个电子邮件地址、一个文件，甚至是一个应用程序。而在一个网页中用来超链接的对象，可以是一段文本，或者是一个图片、图像、动画等媒体。当浏览者单击已经链接的文字或图片后，链接目标将显示在浏览器上，并且根据目标的类型来打开或运行。

一个网站包括多级网页层次结构，如主页、次主页（主要是一些主要栏目）和普通网页，它们共同构成一个多层次的网站结构。访问网站时，首先看到的是一个网站的主页，而在网站主页中又会把下一级的次主页以超链接的方式显示出来，这样用户就可以从网站主页上直接打开次主页，在次主页下面又可以有次次主页。因为一般对外公告的只是网站域名，即默认主页，因此，如果在主页中没有把次主页的链接给出来，用户就不知道如何打开这些次主页了。

当然，每级主页下面又会有许多普通的网页，这些网页也会在对应该级别的主页中显示出来，或者通过网页中的一些关键字超链接进入到具体的网页，毕竟再多的各级主页也不可能把全部的网页都以超链接的方式显示出来。通常，在各级主页中仅显示最新的普通网页超链接。

11.2.6 HTTP服务访问基本流程

HTTP是一个面向文本（text-oriented）的应用层协议，所使用的服务端口是TCP的80端口（这就是HTTP服务的传输层地址），通信双方就是在这个端口上进行通信的。当然，这个端口是可以更改的，但目前互联网上的Web服务器都是使用这个默认的80端口。每个Web服务器都有一个应用进程，时刻监听着80端口的用户访问请求。当有用户请求（以HTTP请求报文方式）到来时，它会尽快做出响应（以HTTP响应报文方式），返回用户访问的页面信息。当然，这一切都建立在传输层创建好对应的TCP连接基础之上。

至于何时关闭TCP连接，则要视所使用的HTTP版本而定。在HTTP/1.0及以前版本中，经历每一个请求-应答过程后就会关闭所使用的TCP连接，单击新的链接后又会重新建立新的TCP连接。这种HTTP服务方式称为非持续连接（no-persistent connection）。而在HTTP/1.1版本以后，HTTP允许在同一个TCP连接基础上访问同一网站服务器上的多个不同页面，仅当用户关闭对应的网站时，对应的网站TCP传输连接才关闭，称为持续连接（persistent connection）。持续连接模式的Web服务访问的基本流程如图11-4所示。

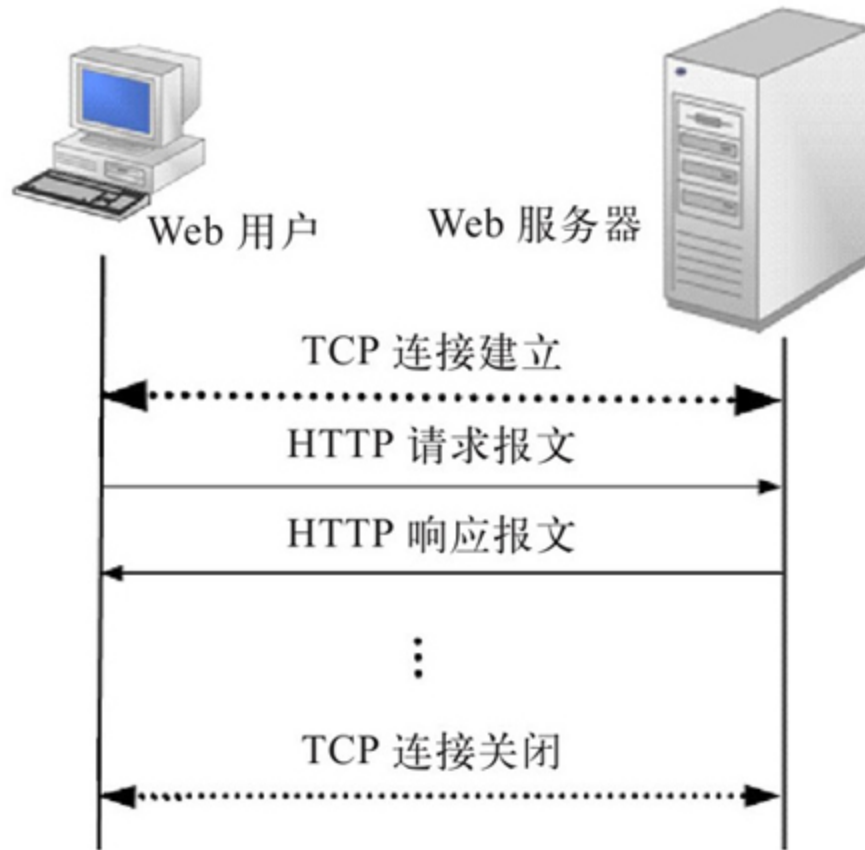


图 11-4 持续连接模式的Web服务访问的基本流程

从图11-4中可以看出，HTTP服务的访问主要是指HTTP消息的传递，其中包括Web用户向Web服务器提交的HTTP请求报文和Web服务器向Web用户返回的HTTP响应报文。下面先综合了解一下HTTP，然后再具体了解其报文结构。

11.2.7 HTTP的主要特性

HTTP是一个属于应用层的面向对象的协议，由万维网协会（W3C）和Internet工作小组（Internet Engineering Task Force, IETF）于1990年合作开发。1997年前使用的是RFC1945中所定义的HTTP/1.0版本，1998年后使用的是其更新版本RFC2616，即HTTP/1.1版本。HTTP规定了Web客户端与Web服务器之间的报文交互方式，其报文中的每个字段都是一些ASCII字符串，而且各个字段的长度都是不固定的。

总体来说，HTTP具有以下几方面的特性：

（1）客户端/服务器（C/S）模式

C/S模式是所有网络应用服务采用的通用模式。Web客户只需要使用支持HTTP的客户端程序（各种浏览器），就可以访问由不同Web服务器程序开发的Web网站。

（2）无连接

这里所说的“无连接”是指在进行Web应用前无须建立专门的HTTP应用层会话连接，仅需要直接利用传输层已为它建立好的TCP传输连

接即可，而像Telnet、SMTP、POP3这类应用协议，是面向连接的，除了需要传输层的TCP连接外，它们自己还要建立会话连接。

(3) 高可靠性

虽然HTTP本身是不可靠的无连接协议，但它使用了可靠的TCP传输层协议，在进行HTTP传输之前，已建立了可靠的TCP连接，因此，从数据传输角度来讲，HTTP的报文传输仍是可靠的。

(4) 无状态

这里所说“无状态”是指同一客户第二次访问同一Web服务器上的同一页面时，服务器给客户端的响应与第一次是一样的（当然，这是假设Web服务器上的对应页面没有更新），Web服务器不会记住这个客户端曾经访问过这个页面，而做出任何其他响应。

(5) 简单快速

客户端通过HTTP访问Web服务器时，只需传送请求方法和路径。请求方法常用的有GET、HEAD、POST等（具体内容将在本章后面介绍），每种方法规定了客户端与服务器之间进行的事务处理和消息类型。由于HTTP简单，使得HTTP服务器的程序规模小，因而通信速度可以很快。

从图11-4可以看出，**HTTP**报文分为请求报文和响应报文两类，分别用于**Web**客户端发起的**HTTP**应用请求和**Web**服务器对客户端所做出的**HTTP**应用响应。下面将分别介绍这两种**HTTP**报文格式。

11.2.8 HTTP请求报文格式

在建立好TCP传输连接后，Web客户端首先要进行的是向Web服务器发送HTTP请求报文，请求打开指定的网站或页面。一个HTTP请求报文包括请求行（request line）、请求头部（request header）行、空行和实体主体（entity body）行4个部分，如图11-5所示。HTTP报文

（HTTP请求报文和11.2.9节将要介绍的HTTP响应报文）中各字段是没有固定长度的。下面对这4个组成部分分别进行说明。

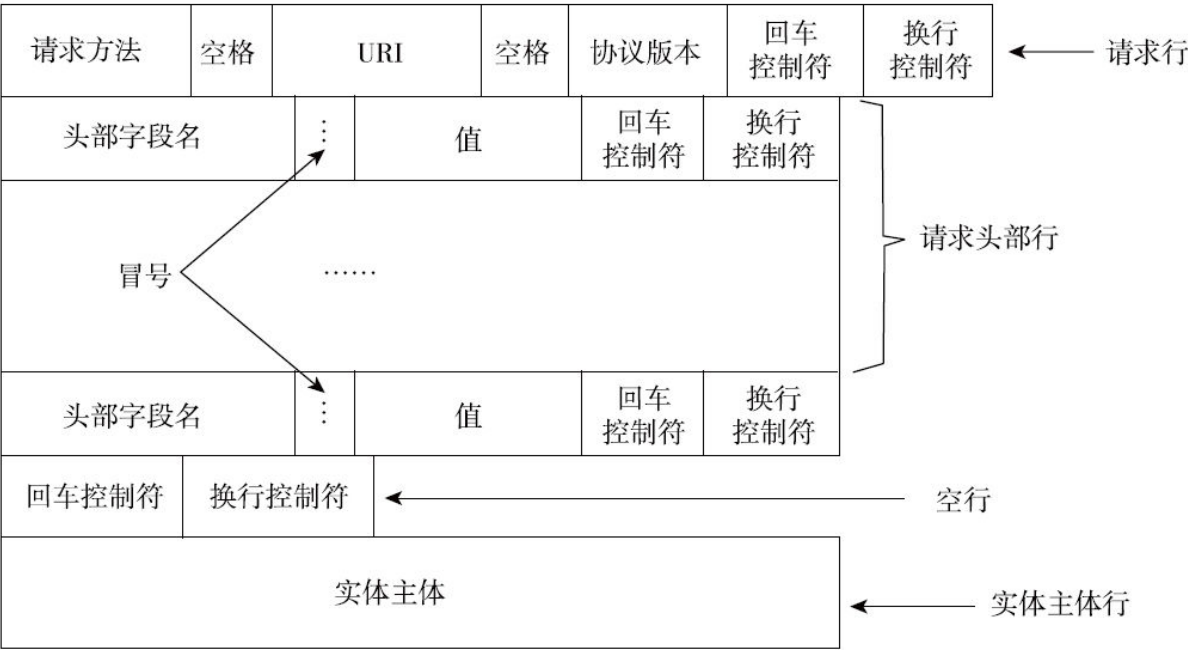


图 11-5 HTTP请求报文格式

1.请求行

HTTP请求报文中的“请求行”是由“请求方法”、“URL”和“协议版本”3个字段组成的，它们之间均以空格进行分隔。这部分是必不可少的。在请求行的最后有一个回车控制符和一个换行控制符（一起以“CRLF”表示），使下面的请求头部信息在下一行显示。

“请求方法”字段指示本请求报文中所使用的HTTP操作（其实就是指所使用的HTTP命令），具体如表11-2所示（注意，全是大写字母，不能为小写）。其中最常用的是GET和POST这两种方法。

表 11-2 HTTP 请求方法

请求方法	含义说明
GET	请求服务器发送在 URL 字段中指定的 Web 页面。URL 的根部分是相对服务器的根目录的，总以 “/” 前缀开始，下同
HEAD	请求读取 URL 字段指定的 Web 页面的头部信息，而不是全部的 Web 页面。利用这一方法可以得到一个页面的最后修改时间或者其他头部信息
PUT	请求在 URL 路径下存储一个 Web 页面，与 PUT 方法相反
POST	在 URL 所指定的 Web 服务器后面附加一个以 URI 格式命名的资源，以便可以为 Web 服务器提供更多的信息，如在服务器上张贴一条海报消息

(续)

请求方法	含义说明
DELETE	删除 URL 字段指定的 Web 页面，当然最终能否删除成功还会受到用户权限的限制
TRACE	指明这是一个用来进行环回测试的请求报文，用于调试
CONNECT	用于连接代理服务器
OPTIONS	请求查询一些特定的选项信息

例如，“GET http://winda.blog.51cto.com/55153/815617 HTTP/1.1”是请求打开作者在51CTO博客上的一篇文章，其中GET是请求方法，中间部分是URL，HTTP/1.1是使用的HTTP版本，且它们三者之间用空格分隔。

2.请求头部行

HTTP请求头部包括一系列的“请求头”和它们所对应的值，指出允许客户端向服务器传递请求的附加信息以及客户端自身的信息。当打开一个网页时，浏览器要向网站服务器发送一个HTTP请求头，然后网站服务器根据HTTP请求头的内容生成当次请求的内容并发送给浏览器，这就是HTTP请求报文中的“请求头”的作用。

HTTP请求报文的“请求头部”是由一系列的行组成的，每行包括“头部字段名”和“值”两个字段，它们之间用英文冒号“:”分隔，但也可以没有“请求头部”。每一行的最后都有一个回车控制符和一个换行控制符（一起以“CRLF”表示），使下一个请求头在下一行显示。许多请求头都允许客户端指定多个可接收的“值”字段选项，有时甚至可以对这些“值”字段选项进行排名，多个选项间以逗号分隔。典型的HTTP请求头如表11-3所示。

表 11-3 典型的 HTTP 请求头

请求头	含义说明
Accept	指定客户端能处理的 MIME（Multipurpose Internet Mail Extension，多用途互联网邮件扩展）页面类型。例如，Accept: image/gif，表明客户端希望接受 GIF 图像格式的资源；Accept: text/html，表明客户端希望接受 HTML 文本
Accept-Charset	指定客户端可以接受的字符集。例如，Accept-Charset:iso-8859-1,gb2312，表示客户端可以接受的字符集在所列的两个标准中。如果在请求消息中没有设置这个请求头，默认是任何字符集都可以接受
Accept-Encoding	指定客户端能进行解码的数据编码方式。例如，Accept-Encoding:gzip, compress，表示客户端可以接受 gzip 和 compress 两种编码格式。如果请求消息中没有设置这个请求头，那么服务器假定客户端可以接受各种内容编码
Accept-Language	指定客户端能处理的语言类型。例如，Accept-Language:zh-cn，表示客户端只接受中文简体。如果请求消息中没有设置这个请求头，那么服务器假定客户端对各种语言都可以接受
Authorisation	指定客户端信任的凭据列表。当浏览器访问一个页面时，如果收到服务器的响应代码为 401（未授权），那么可以发送一个包含 Authorization 请求报头域的请求，要求服务器对所列的用户账户进行验证
Cookie	将一个以前设置的 Cookie 送回给服务器
Connections	请求采用持续连接方式。例如，Connection:Keep-Alive
Date	指定消息发送时的日期和时间。例如，Date: Tue, 11 Jul 2012 18:23:51 GMT

(续)

请求头	含义说明
From	向服务器请求一个电子邮箱地址。例如，From: winda@microsoft.com，这是假设作者在访问微软网站时向微软的服务器申请的一个电子邮箱地址
Host	指定被请求服务器的域名和端口，如果使用默认的 80 端口，则可以省略端口指定步骤。例如，Host: www.5lcto.com，表示请求的服务器域名为 www.5lcto.com，且使用默认的 TCP 80 端口进行访问
Referer	包括一条 URI 信息，要求从指定的 URI 中访问当前请求的页面
User-Agent	允许客户端将它的操作系统、浏览器和其他属性告知服务器。例如，上网登录论坛的时候，往往会看到一些欢迎信息，其中列出了登录者的操作系统的名称和版本，以及他所使用的浏览器的名称和版本，这就是 User-Agent 请求头的“功劳”了
Upgrade	客户端希望切换到新的协议

3.空行

在HTTP请求报文的最后一个请求头后是一个空行，发送回车符和换行符（一起以“CRLF”表示），通知服务器以下不再有请求头。

4.实体主体行

请求报文中“实体主体”部分通常是不用的。它不能在GET方法中使用，仅在POST方法中用于向服务器提供一些用户凭据信息。

下面是一个HTTP请求报文示例：

```
POST/servlet/default.jsp HTTP/1.1 !--- 这是一个POST方法请求行，指出了
所附加的消息的URL和所用的HTTP版本为1.1
Accept-Language:zh-cn,zh !--- 指定客户端可以接受的语言
Accept-Charset:GB2312,utf-8 !--- 指定客户端可以接受的字符集为GB2312和
utf-8
Accept:text/html,application/xhtml+xml,application/xml !--- 指定客
户端浏览器支持的MIME类型分别是text/html、application/xhtml+xml、
application/xml
Accept-Encoding:gzip,deflate !--- 指定客户端浏览器支持的压缩编码格式为
gzip和deflate
User-Agent:Mozilla/5.0 !--- 指定客户端使用的浏览器为Mozilla 5.0版本
Host:www.51cto.com !--- 指定客户端要访问的服务器域名为www.51cto.com
Connection:Keep-Alive !--- 指定采用持续连接方式
(这里有一个空行)
user=winda&pwd=1234 !--- 此行为向服务器提交的用户账户为winda，密码为
1234
```

11.2.9 HTTP响应报文格式

Web服务器收到客户端发来的HTTP请求报文，通过服务器处理后会返回一个HTTP响应报文给请求的客户端，以告知客户端Web服务器对客户端请求所做出的处理。如是否允许此次请求的HTTP连接，出现了什么连接错误，错误代码和错误原因等。HTTP响应报文也是由四部分组成，分别是响应行（Response）、响应头部（Response Header）行、空行和实体主体行，如图11-6所示。

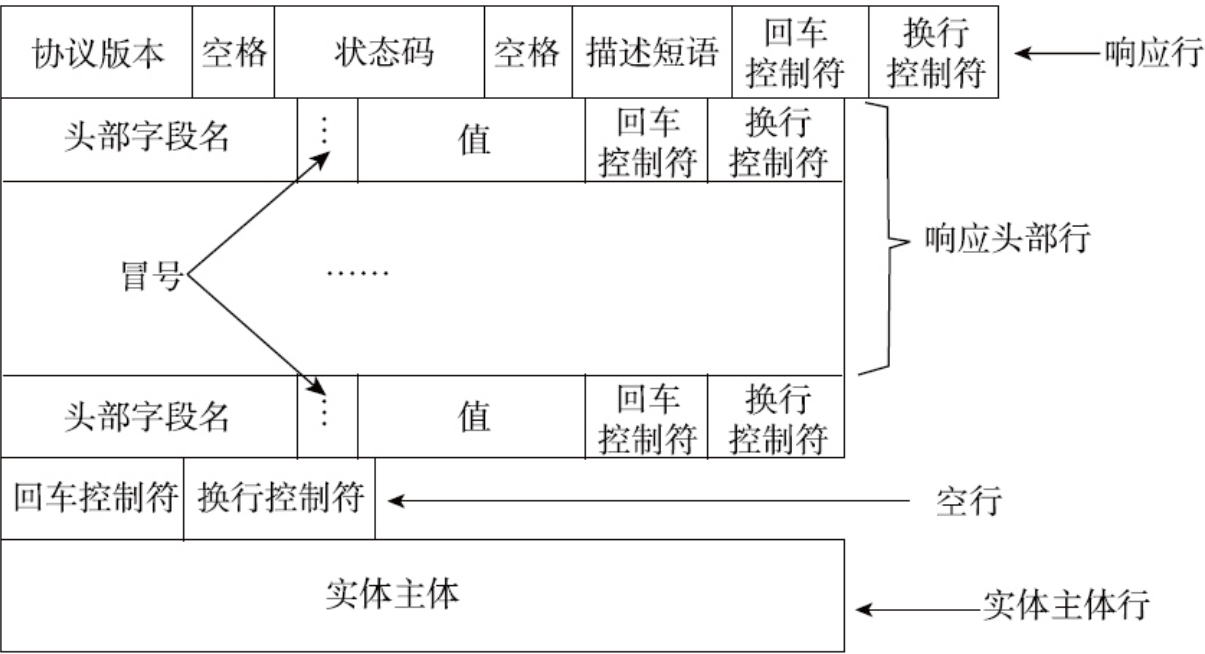


图 11-6 HTTP响应报文格式

HTTP响应报文中的“空行”部分只有一个回车控制符和一个换行控制符，其目的就是空一行显示下面的主体信息。“实体主体行”部分也

基本不用。下面分别介绍“响应行”和“响应头部行”。

1.响应行

在HTTP响应报文的“响应行”中主要有3个字段，分别是“协议版本”、“状态码”和“描述短语”，它们之间用空格分隔。最后还有一个回车控制符和一个换行控制符（一起用“CRLF”表示）。其中“协议版本”字段用来显示Web服务器使用的HTTP版本。“状态码”字段比较重要。它用一个3位数表示不同的状态，如请求是否被接受，如果没有被接受原因是什么，共有5组取值，如表11-4所示。“描述短语”字段是对应状态码的简短描述。

表 11-4 HTTP 响应报文中的“状态码”类型

状态码类型	含义说明	示例
1xx	指示类响应，表示请求已接收，继续处理	100= 服务器同意处理客户请求
2xx	成功类响应，表示请求已被成功接受	200= 请求成功；204= 无内容，也表示请求成功
3xx	重定向类响应，表示要完成请求必须进行更进一步的操作	301= 指示页面已重定向了
4xx	客户端错误类响应，表示客户端请求有语法错误或请求无法实现	400= 客户请求有语法错误，不能被服务器所理解；401= 请求未经授权；403= 服务器收到请求，但拒绝提供服务；404= 请求资源不存在
5xx	服务器端错误类响应，表示服务器未能实现所需的请求	500= 服务器发生了不可预期的错误；503= 服务器当前不能处理客户端的请求，过一段时间再试

“响应行”的最后也是一个回车控制符和一个换行控制符，然后换行进入到下面的“响应头部行”部分。

2.响应头部行

HTTP响应报文中的“响应头部行”允许服务器传递不能放在状态行中的附加响应信息，以及关于服务器的信息和对URI所标识的资源进行下一步访问的信息。

HTTP响应报文的“响应头部行”由一系列行组成，每行包括“头部字段名”和“值”两个字段，它们之间用英文冒号“:”分隔。每一行的最后都有一个回车控制符和一个换行控制符，使下一个响应头在下一行显示。许多响应头都允许服务器指定多个返回的“值”字段选项，有时甚至可以对这些选项的“值”字段选项进行排名，多个选项间以“/”分隔。典型的HTTP响应头如表11-5所示。

表 11-5 典型的 HTTP 响应头

响应头	含义说明
Allow	显示服务器支持哪些请求方法
Server	显示服务器的软件信息。例如，Server：Apache-Coyote/1.1，表示 Web 服务器软件为 Apache-Coyote，版本为 1.1
Content-Encoding	显示请求文档采用的编码方法。例如，Content-Encoding：gzip，表示文档采用的是 gzip 压缩方式
Content-Language	显示请求页面使用的语言。例如，Content-Language：zh/zh-cn，表示页面支持的语言
Content-Length	显示请求页面的长度（以字节为单位）。例如，Content-Length：1024，表示页面大小为 1024 字节
Content-type	显示页面支持的 MIME 类型。例如，Content-type：text/html，表示页面支持 text 和 HTML 两种消息类型
Date	指定消息发送时的日期和时间。例如，Date：Tue, 11 Jul 2012 18:23:51 GMT
Last-Modified	显示请求页面最后被编辑或修改的日期和时间。例如，Last-Modified：Tue, 11 Jul 2012 12:11:51 GMT
Location	指示客户端将请求发送到指定的位置，起到重定向的作用
Accept-Range	显示服务端将接受指定字节范围内的请求。例如，Accept-Range：1024，表示服务器只接受 1024 字节以内的请求
Refresh	指示客户端多少秒后再刷新或者重新访问本页面。例如，Refresh：5，表示要客户端过 5 秒后再刷新或访问本页面
Set-Cookie	指示客户端设置和页面关联的 Cookie
Upgrade	服务器希望切换到新的协议

下面是一个HTTP响应报文的示例。

HTTP/1.1 200 OK !--- 这是一个HTTP响应报文状态行，显示的协议版本为HTTP/1.1，表示请求成功
Server:Microsoft-IIS/6.0 !--- 表示服务器
Date:Sun,3 Jan 2012 13:13:33 GMT !--- 表示服务器发送此响应报文的日期和时间
Content-Type:text/html !--- 指示所请求页面支持的MIME类型为text和HTML
Last-Modified:Mon,14 Apr 2012 13:23:42 GMT !--- 指示所请求页面的最后修改日期和时间
Content-Length:112 !--- 指示服务器接受前112个字节的请求内容

11.3 DNS服务

DNS（Domain Name System，域名系统）是一种把计算机主机名称解析为对应的IP地址的服务。在UNIX和Linux操作系统中的DNS服务通常称为BIND（Berkeley Internet Name Domain Service，伯克利因特网名称域服务）。但要特别说明的是，从网络通信原理上来讲，DNS并不是必需的，因为可以直接通过IP地址进行访问，而且事实上网络通信时最终所采用的寻址方式也是网络层的IP地址寻址。下面先来介绍一下DNS技术的引入背景。

11.3.1 DNS技术的引入背景

为什么要引入DNS服务？首先让我们回顾一下数据通信的基本原理。

在介绍传输层服务时，我们提到过应用层的Socket（套接字）服务。套接字是IP地址和端口号的组合，中间用冒号或逗号分隔，如（192,168.10,80）。每个TCP传输连接有两个端点，也就是有两个套接字，即“源IP地址和端口号”组合和“目的IP地址和端口号”组合，可表示为 { socket1,socket2 }，或者 { (IP1,Port1)，(IP2,Port2) }。其中的“IP地址”是主机的网络层地址，用来在局域网或广域网中对各节

点进行唯一标识（当然，局域网和广域网可使用的IP地址范围是不一样的，具体参见本书第7章有关局域网IP地址和公网IP地址的相关内容），而“端口号”则是为每个应用进程所分配的传输层地址。在同一时间内，两台主机间可以运行多个网络应用，每个进程需要用一个端口来进行唯一区分（因为此时多个应用进程中的源IP地址和目的IP地址都是一样的）。

1.IP地址标识的不足

在应用层要访问目的主机时，不仅要指出该目的主机的网络层IP地址（用于网络寻址），还要指出在传输连接中使用的端口号，以及在应用层使用的应用协议。例如，在访问Web服务器时，需要在用户浏览器地址栏中输入类似http://118.144.78.54:80的地址，其中，http是访问Web服务器时使用的应用层HTTP，118.144.78.54是对应Web服务器的IP地址，80是对应Web服务器应用进程传输层端口号（当然，如果是常规服务，那么这个端口号是不用输入的）。

从上述内容可以看出，使用IP地址来运行各种网络应用是完全没问题的。但对于那些位于互联网上，供全球用户公开访问的各种公用服务器来说，这种IP地址标识方式存在以下几方面的不足：

（1）不便记忆

尽管现在把IP地址以比较简单的十进制表示，在浏览器地址栏中也是以十进制格式表示IP地址的，其长度也不是很长（十进制IPv4地址最长也仅为12位），但是它远没有普通的名称好记，就像要记一个人的身份证号要远比记一个人的名字困难一样，哪怕身份证号只有短短的几位。

当然，以上是针对现在仍广泛使用的IPv4地址来说的，对于长度为现行IPv4地址4倍的IPv6地址，这方面的不足就更为明显了。

（2）不方便地址变更

Web服务器，特别是互联网上Web服务器的IP地址可能会因各种原因而变更。如果采用IP地址进行标识的话，那么IP地址的每次变更对于这种开放型的互联网服务器来说可能是致命的打击，因为有那么多已知或未知的用户，不可能一一都通知到。但如果采用的是网站服务器名称来进行标识，那么IP地址如何变化都没有影响，只要服务器主机名称不变即可。

（3）不安全

如果直接把网站服务器的IP地址对外暴露，那么这显然不是一个安全的做法，因为这很可能被一些别有用心的人利用（窃取公网IP地址、盗链接等）。尽管对一些专业黑客来说，无论如何做好安全防范

工作，要想获取服务器IP地址都是一件容易的事，但防范总好过不防范。

2.host文件名称解析方案

为了克服以上IP地址标识方案的不足，于是有人想到使用更方便记忆的服务器名称来替代服务器IP地址。在引入DNS（1987年引入）之前的ARPANet（互联网的前身）中，通过一个称为host.txt的文本文件把网络中各计算机的计算机名称与其对应的IP地址一一列出，以此实现解析。在host文件的IP地址和计算机名映射表项中，前面是IP地址，后面是对应的计算机主机名（最初仅是像我们通常所说的NetBIOS格式计算机名），每个映射独占一行。格式如下（以下仅为示例，并非真实的映射）：

```
127.0.0.1 localhost !--- 此为本地计算机环路映射
196.168.1.109 microsoftdesk
202.16.83.15 sunfinal
...
```

然后，由各计算机调用这个host文件以实现从计算机名到IP地址的解析。这个host文件当时是由专门的一台或多台服务器进行管理的（其实，在Windows、Linux和UNIX系统主机上也有这个文件，仅用于本地计算机上的名称解析）。当然，这是在没有使用DNS域名情况下的映射（主要用于局域网中），在使用了DNS域名时，后面的计算机名也可以是DNS域名格式。

随着互联网上计算机的增加，人们发现以上这种通过host文件进行名称解析的方案至少存在以下两方面的不足：

(1) 名称冲突在所难免

因为没有一种可以限制名称冲突的机制，所以各公司在为自己的服务器起名时就难免与其他公司的服务器计算机名发生冲突。就像每个人的姓名一样，在全国范围来讲，同名的人不知有多少，因为没有规定不允许同名。

(2) 少数名称解析服务器难以承受

随着互联网上服务器数的飞速增加，仅靠授权机构中少数通过调用host文件进行名称解析的名称服务器根本无法承受如此大的负荷。

3.DNS名称解析方案

为了解决以上host文件名称解析方案中的不足，人们开发了基于域（这里的“域”是指“域名”）的分层命名方案——DNS。DNS可以将主机名映射成对应的IP地址，其基本的名称解析原理如下。

当一个应用进程需要把主机名解析为IP地址时，首先该应用程序就调用一个解析器（resolver），使它成为DNS客户，将该主机名作为DNS请求报文参数，以UDP用户数据报方式发送给本地DNS服务器。然后，本地DNS服务器查找该主机名，并且将找到的IP地址放在响应

报文中返回给解析器。解析器再将**IP**地址返回给调用解析器的应用进程，这样应用进程就可以根据所得到的目的**IP**地址进行通信了。当然，可能会存在本地**DNS**服务器不能解析所请求的主机名的情况，这时本地**DNS**服务器就仅为一个**DNS**代理角色，然后它可以把这个解析工作交给在本地服务器上配置的其他相关联的**DNS**服务器。以上就是**DNS**服务器的“转发器”功能。

11.3.2 DNS命名方案的设计思想

从设计思想上来看，开发DNS命名方案就是想通过某种方法解决host名称解析方案中“容易发生名称冲突”和“服务器解析性能不足”两方面的问题。这个方案是采用基于域名的分层命名方案。下面具体分析这一命名方案是如何解决这两方面问题的。

1.名称冲突的解决方案

目前，计算机名主要有两种格式，一种是像早期host文件中那样的非分级的单一字符串所代表的本地计算机名称，如webserver；另一种是以DNS域名为基础的域网络中的DNS名称，如webserver.lycb.com，后面的“lycb.com”是一个域名。完整的DNS名称（webserver.lycb.com）代表的是在lycb.com域中的一台名为webserver的服务器。

注意 其实，DNS域名的最后面还有一个小圆点，代表的是根域名，只是平时我们在书写时不写而已。另外，DNS域名不仅应用于Web网站标识，而且应用于FTP站点和电子邮件系统，只不过具体使用时前面所指定的协议或格式是不一样的：Web网站使用的是HTTP（http://www.itct.com.cn），FTP站点使用的是FTP（如

ftp://www.itct.com.cn)，电子邮箱地址格式是“邮箱用户账户名@邮箱DNS域名”（如winda@itct.com.cn）。

从以上介绍可以看出，DNS名称是由两大部分组成的，最前面的部分代表的是服务器的本地计算机名称，后面全部是对应服务器所在域的域名，而且这两部分中间是以小圆点（.）分隔、连接的，即“本地计算机名.DNS域名”。这就是为什么说DNS是基于域命名方案的原因。

DNS域名不是单一结构，而是包含至少两个层次的分级结构，因为单一结构的域名无法满足全球如此多互联网用户的需求，就像尽管没有限制姓名长度，而且汉字有上万个，但还是很难避免完全不重名。如果真有制度限制不能重名的话，则后面出生的人名字可能都非常难记，或者都是他们自己不喜欢的了。DNS域名也一样，如果只有一级结构，则后面用户所能申请的域名基本上都会很长，且不能符合用户特征要求。

在分层结构的DNS域名中，必须有一个是顶级域名，然后在这个顶级域名下面再申请，或者注册二级、三级，甚至更多级别的域名，各级域名间同样是以小圆点（.）分隔、连接的。最右边的部分代表的是顶级域名，左边部分代表的是子域名，而且是级别最低的域名写在最前面，级别最高的域名写在最后面。下级域名必须隶属于上级域名。例如，itct.com.cn这个DNS域名是一个三级域名，其中itct是三级

域名（在本域名中级别最低），它隶属于其二级域名com，而二级域名com又隶属于其一级域名（又称顶级域名）cn。在每个级别域名下，又可以有多个并列关系的下级域名，如顶级域名cn下面可以有com、net、gov、edu等二级域名，而在net这样的二级域名下面同样可以有由各用户申请注册的三级域名。

设计这样的分层结构域名的好处很明显，就是上级域名下面可以有多个不同的下级域名，且整个DNS域名中只要有一级域名不一样，就可认为该域名与其他域名是不一样的。这样就可以比较好地确保互联网上域名的唯一性。例如，mov.microsoft.com和mail.microsoft.com这两个域名就是不同的域名，sina.com.cn和sina.com也是两个不同的域名。

2.解析性能不足的解决方案

在早期利用host文件保存计算机名到IP地址映射的解决方案中，网络中的名称解析工作全是由少数几个集中存储host文件的服务器主机来担当的。随着网络规模的不断扩大，这种完全集中的管理方式显然不能满足解析性能要求，在今天的互联网上更是不可能实现。

有了DNS的分层命名方案后，就可以在host文件集中管理的基础上进一步采取分布式的管理方式。也就是说，可以针对每一级域名用少数的DNS服务器单独进行名称解析，这样一来，各DNS服务器的解

析压力就不会变得难以承受了。例如，在DNS名称www.microsoft.com中，顶级域名com是由对应的顶级域名提供商的DNS服务器负责解析的，二级域名microsoft是由对应的二级域名提供商的DNS服务器负责解析的。

目前，整个互联网上可以注册的域名非常多，而且每一类域名又可以有许多域名提供商提供，这样一来，即使是同一类域名，在全球范围内都可以有非常多分布在全球不同国家或地区的域名提供商的DNS服务器来共同担当名称解析服务，不会存在性能不足的问题。但要特别说明的一点是，这些DNS服务器都不是孤立的，即使是完全不同的域名解析服务器，它们之间都有依次的从属关系，这与域名提供商的级别有关。下级域名提供商的DNS服务器必须与上级域名提供商的DNS服务器建立某种关联，否则可能同一域名会在不同地区被多人注册，这是不允许的。

11.3.3 DNS名称空间

从大的范围上讲，整个互联网是一个**DNS**名称空间，这个空间是由当时可以注册的各级域名共同构成的。当然，这个空间会随着用户的需求和互联网域名管理机构（**ICANN**或**NeuLevel**）颁布的可注册域名数量和类型的扩展而扩展。整个**DNS**名称空间像一棵倒过来的树

（如图11-7所示），最顶端称为互联网的“根域”，以一个小圆点（.）表示，接下来是顶级域，再接下来是二级域、三级域，依此类推（一般是三级结构。注意，域名是不区分大小写的，但通常是小写的）。其中，顶级域名和二级域名必须是由对应的互联网域名管理机构颁布的。

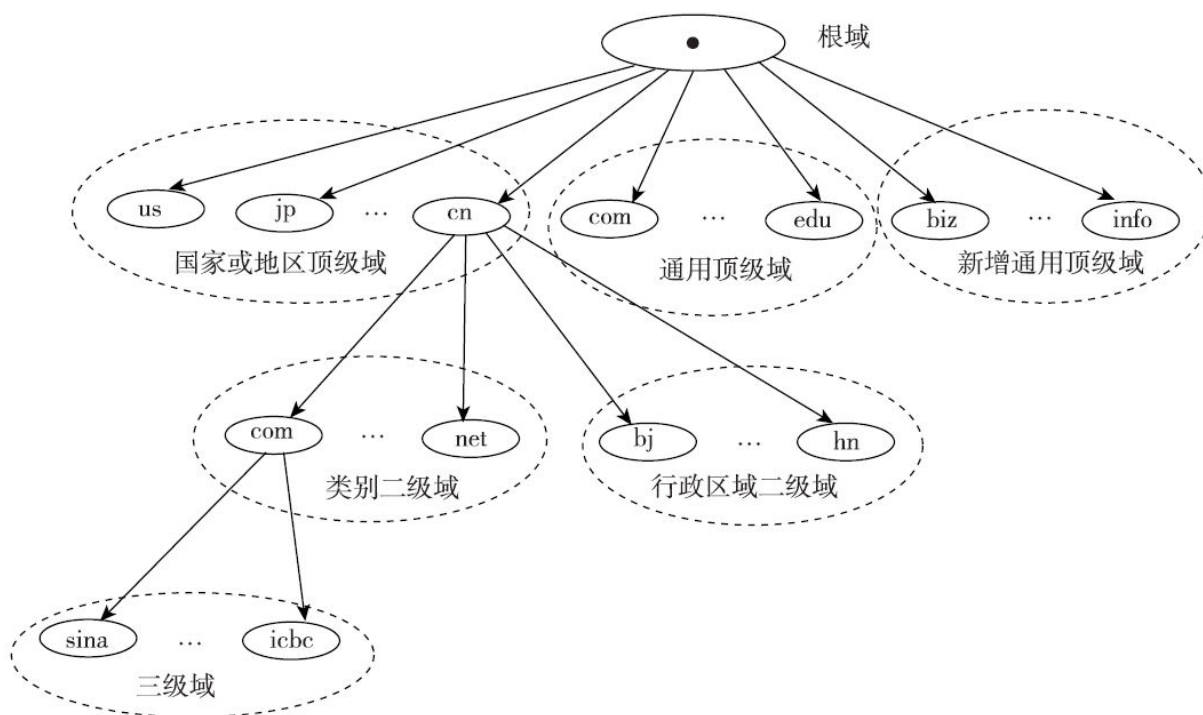


图 11-7 互联网上的DNS名称空间

目前，互联网上注册的DNS域名主要有以下3个级别：顶级域名、二级域名和三级域名，其中只有二级及其以下级别的域名才可由用户自己申请，用户不能直接申请顶级域名。

1. 顶级域名

顶级域名是根域名（.）下面的第一级域名，但它不能单独为用户分配。目前顶级域名有三大类：第一类是国家或地区类，第二类是通用类，第三类是近期新增的通用类。国家或地区类顶级域名是为每个国家或地区分配的域名，由ISO3166进行规定，如cn代表中国，us代表美国，jp代表日本.....

通用类顶级域名在RFC1591中进行规定，如商业类为com、教育类为edu、政府类为gov、国际组织类为int、网络供应商类为net、非盈利性组织类为org.....近期新增的通用类顶级域名有biz（商业组织）、film（公司企业）、store（商店）、web（突出WWW活动的组织）、arts（突出文化、娱乐活动的组织）、rec（突出消遣、娱乐活动的组织）、info（提供信息服务的组织）、.nom（个人）。其中，biz顶级域名是为了接替com域名的，因为com域名空间已消耗得差不多了，但它不再是由ICANN管理，而是由NeuLevel公司单独管理。

2. 二级域名

二级域名是指顶级域名之下的域名。它又分为两大类：在国际顶级域名下，它是指域名注册人的网上名称，例如ibm、yahoo、microsoft等，这是可供用户申请的；在国家或地区顶级域名下，它表示注册企业类别的符号，如com、edu、gov、net等，这些是不能直接供用户申请注册的。我国在国际互联网络信息中心（Inter NIC）正式注册并运行的顶级域名是cn，这也是我国的一级域名。目前，我国有中文域名，其中一级域名为“中国”。

在顶级域名之下，我国的二级域名又分为类别域名和行政区域名两类。类别域名共6个，包括用于科研机构的ac、用于工商金融企业的com、用于教育机构的edu、用于政府部门的gov、用于互联网络信息中心和运行中心的net、用于非盈利组织的org。行政区域名有34个。

3.三级域名

三级域名也是可以由用户自己申请注册的，可以采用字母（A~Z、a~z、大小写组合等）、数字（0~9）和连接符（-）等，各级域名之间用小圆点（.）连接，三级域名的长度不能超过20个字符。如无特殊原因，建议采用申请人的英文名（或者缩写）或者汉语拼音名（或者缩写）作为三级域名，以保持域名的清晰性和简洁性。

一般来说，企业的互联网域名有三级就可以了，如果还需要扩展，可以继续申请更多级别的域名。三级及其以下级别的域名是可以

自己申请注册的，只要在其上一级域名下没有重复的域名即可。另外，以上是针对整个互联网DNS域名来说的，其实针对具体的公司互联网DNS域名，同样有一个名称空间，那就是由所申请的顶级域名下扩展的各级域名共同构成的名称范围（如图11-8所示的是“新浪中国”的一部分DNS名称空间示例），其中至少包括了五级域名，当然，要能在互联网供用户访问，各级域名也都必须向互联网域名管理机构申请、注册。

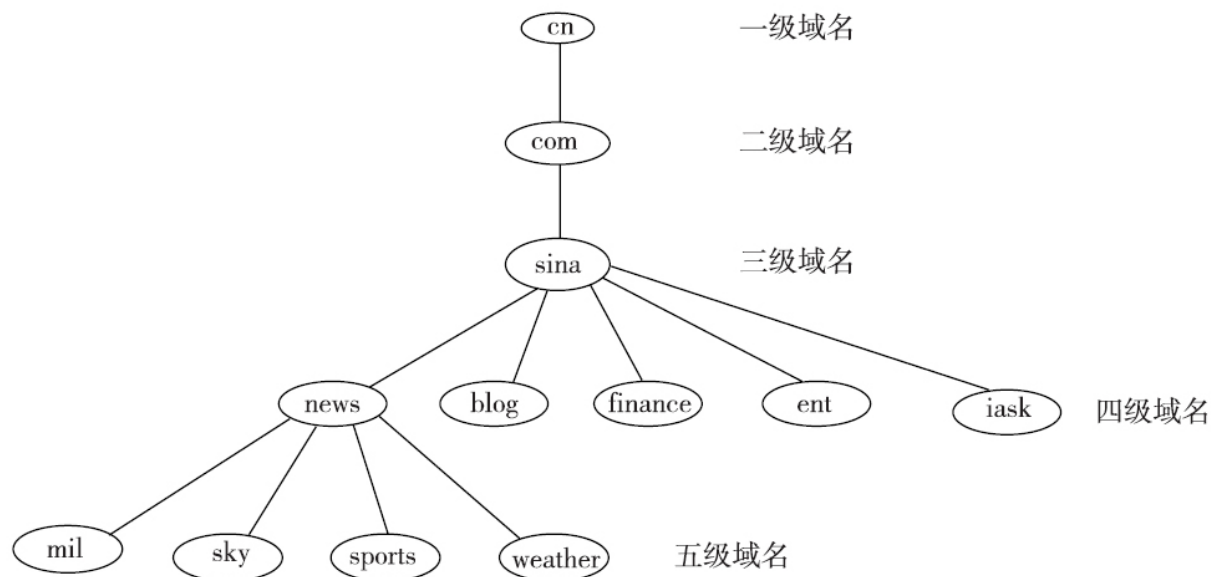


图 11-8 新浪中国的部分域名空间

经验之谈 申请的DNS域名是不包括NetBIOS计算机名部分的，如microsoft.com是微软公司所申请的一个二级域名。在这个域名下，可以有许多不同用途的主机，如Web服务器、FTP服务器、E-mail服务器等，而且它们都可以有多个同类服务器。这时，像

www.microsoft.com、ftp.microsoft.com、mail.microsoft.com之类的DNS名称中的www、ftp、mail是服务器的NetBIOS计算机名，不包括在DNS域名之内。因此，以上DNS名称都属于二级域名，而不是三级域名，当然，如果确实申请了像www、ftp、mail之类的三级域名，则另当别论。无论如何，在浏览器中输入的服务器地址中最左边部分肯定只是该服务器的NetBIOS计算机名，而不是属于DNS域名。

11.3.4 DNS名称服务器

前面介绍的DNS系统仅是从服务器命名角度来讲的一项技术，但真正担当DNS域名解析任务的还是那些配置了DNS服务的服务器，就是DNS服务器，专业名称为“名称服务器”（或者“域名服务器”）。在DNS名称服务器中，要特别注重两个方面：一是DNS服务器的分区管理，二是DNS服务器的不同类型以及它们之间的关系。

1.DNS名称服务器的分区管理

从理论上来说，每一级的每个DNS域名都需要配备一台专门用于名称解析的DNS名称服务器，但是全球有那么多域名，每个域名都专门配备的话，则互联网域名管理机构所需配备的DNS名称服务器就实在太多了，这明显不是一种有效的解决方案。

为了提高每台DNS名称服务器的运行效率和利用率，在DNS技术中为此专门提出了“区域”（zone）的概念，就是可以让一台或多台DNS名称服务器负责一个区域的计算机域名解析，而这个“区域”可以包括一个DNS域名树（或者说“DNS域名空间”）的一部分（通常是包括多级域名）或全部，但是绝对不可能大于一个域名树，也就是不能把几个完全不同的域配置为一个区域，如不能把gz.lycb.com和sh.lycb.net划分为一个DNS区域。DNS区域名是对应区域中公共的域名树部分，在

如图11-9所示的示例中，假设只划分一个区域，则这个区域名只能是lycb.com，因为这才是整个区域中的公共部分。

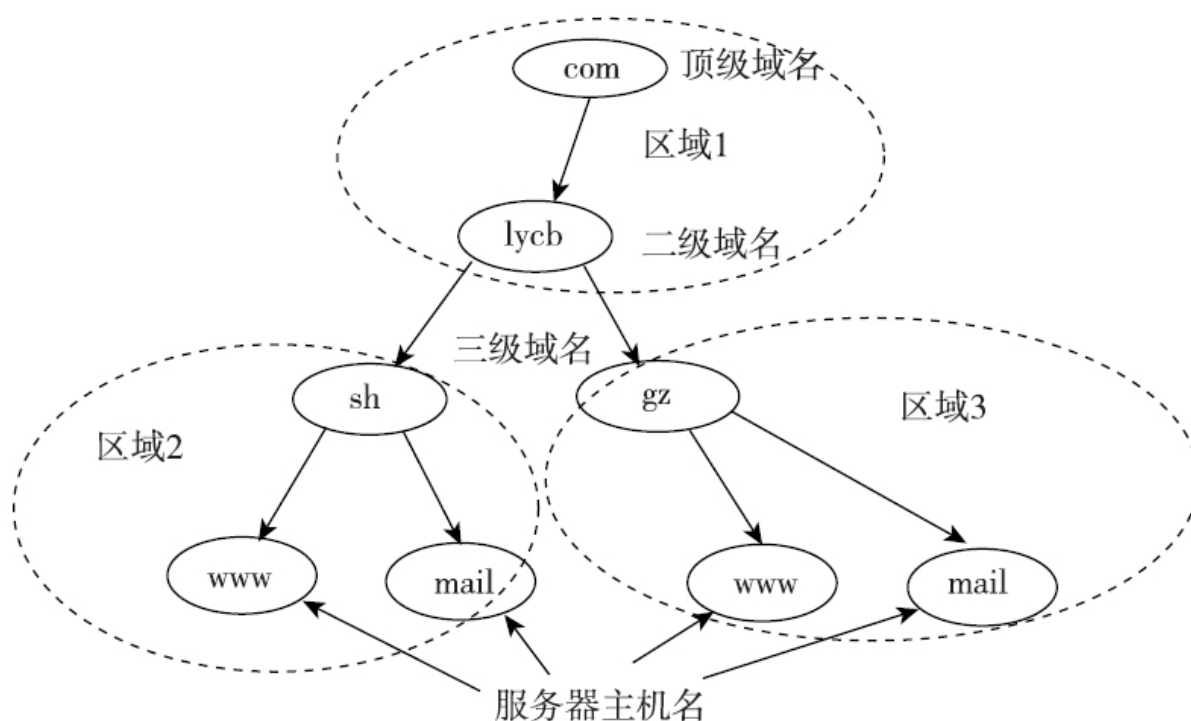


图 11-9 DNS名称服务器分区示例

在如图11-9所示的DNS域名树示例中，如果为每一级的每个域名分别配置一个DNS名称服务器，那么这样一个简单的域名树就需要8台DNS名称服务器，这显然是不合算的，也是没必要的。事实上，如果这个域网络中的应用服务器不是很多的话，整个此DNS域就只划分一个区域（lycb.com），仅用一台或多台DNS服务器负责名称解析工作；如果各级域名中的应用服务器数比较多，则可以将整个DNS域名树分成如图11-9所示的3个区域，区域名分别为lycb.com、sh.lycb.com和

gz.lycb.com，然后为这3个区域各配置一台或多台DNS名称服务器即可。这就是DNS名称服务器的分区理念。

至于具体要把哪几级域名划分在同一个区域，这个不是固定的，可由管理员来决定，但一般取决于各级域名中应用服务器的数量，以及这些服务器的物理位置。通常是尽可能使各区域中应用服务器的数量均衡，以便使各区域中的DNS名称服务器的解析负荷尽可能均衡。同时，还应尽可能把物理位置相近的应用服务器划分到同一个区域中。当然，越高一级别区域的DNS名称服务器的负担就越重，因为它同时还会为下级区域中的DNS名称服务器提供解析支持，即在下级区域中的DNS名称服务器解析不了或者满负荷时提供帮助。

2.DNS名称服务器种类

为了有效地管理整个互联网的DNS域名解析工作，DNS系统开发者设计了一个与分层的DNS域名结构类似的层次化DNS名称服务器结构，把所有DNS名称服务器自高到低分成4个级别：根名称服务器、顶级名称服务器、权威名称服务器和本地名称服务器。它们都是由互联网域名管理机构或下级的ISP负责配置建立的，但它们都不是由一台服务器担当，即使是一个具体的域名服务器也都有一台主服务器和多台辅助服务器，就像在配置Windows Server 2003 DNS服务器时要配置一台主DNS服务器、一台或多台辅助DNS服务器一样。下面分别介绍这几种名称服务器。

(1) 根名称服务器

“根名称服务器”(root name server)是由互联网管理机构配置建立的,是最高层次的名称服务器,负责对互联网上所有“顶级名称服务器”进行管理,有全部的顶级名称服务器的IP地址和域名映射。

全球共有13套(注意,不是13台,每套根名称服务器都有好多台用于负载均衡的根名称服务器,总的根名称服务器据说至少有上千台了)根名称服务器。但每套根名称服务器都只有1个主根名称服务器(也就是说,总共只有13台主根名称服务器),其中10台放置在美国,两台在欧洲,1台在日本。这13台主根名称服务器主机名分别为字母A~M,即完整DNS名称为a.rootserver.net~m.rootserver.net,具体如表11-6所示。其余的均为辅助根名称服务器,分布在世界各地,其目的是为了让世界各地都能就近获取根名称服务器的解析。任何域名解析都要经过这13套根名称服务器获得顶级名称服务器索引。

表 11-6 13 台主根名称服务器的管理机构和 IP 地址

主根名称服务器主机名	管理单位及设置地点	IP 地址
A	INTERNIC.NET（美国，弗吉尼亚州）	198.41.0.4
B	美国信息科学研究所（美国，加利福尼亚州）	128.9.0.107
C	PSINet 公司（美国，弗吉尼亚州）	192.33.4.12
D	马里兰大学（美国马里兰州）	128.8.10.90
E	美国航空航天管理局（美国加利福尼亚州）	192.203.230.10
F	因特网软件联盟（美国加利福尼亚州）	192.5.5.241
G	美国国防部网络信息中心（美国弗吉尼亚州）	192.112.36.4
H	美国陆军研究所（美国马里兰州）	128.63.2.53
I	Autonomica 公司（瑞典，斯德哥尔摩）	192.36.148.17
J	VeriSign 公司（美国，弗吉尼亚州）	192.58.128.30
K	RIPE NCC（英国，伦敦）	193.0.14.129
L	IANA（美国，弗吉尼亚州）	198.32.64.12
M	WIDE Project（日本，东京）	202.12.27.33

其实，在我们配置Windows Server 2003 DNS服务器的“根提示”时就可以见到这些根名称服务器，如图11-10所示，从中也可以见到如表11-6所示的这些主根名称服务器的IP地址。当然，如果配置的是局域网的域名，这些主根名称服务器用不上，因为这些主根名称服务器仅用于互联网上的域名解析。



图 11-10 Windows Server 2003 DNS服务器中的根名称服务器配置对话框

注意 根名称服务器并不直接用于名称解析，因为这些根名称服务器上也没有保存全部的互联网域名记录（仅有负责管理顶级域名的顶级名称服务器的相关记录）。根名称服务器的作用是仅当用户本地名称服务器解析不了某个顶级域名时，告诉本地名称服务器去找哪个顶级名称服务器。

（2）顶级名称服务器

“顶级名称服务器”（top level name server）是各顶级域名自己的名称服务器，负责它们各自所管理的二级域名解析。每一个顶级域名，不论是gTLD（通用顶级域），还是ccTLD（国别顶级域），它们都有自己的域名服务器。如顶级域名com和net有13台名称服务器，主机名也是A~M，域名为gtld-servers.net；顶级域名biz有6台名称服务器，主机名是A~H，域名为gtld.biz。

（3）权威名称服务器

“权威名称服务器”（authoritative name server）是针对前面所说的DNS区域提供名称解析服务而专门配置、建立的名称服务器，可为用户提供最权威的DNS域名解析。每一个域名在互联网上都可找到一台权威名称服务器，ISP也可为用户的每一个DNS域名区域配置一台权威名称服务器。

（4）本地名称服务器

这里所说的“本地名称服务器”不是指用户局域网中的名称服务器，而是用户端操作系统所配置的、由本地ISP提供的名称服务器（也就是本地DNS服务器）。它是离用户最近的互联网名称服务器。用户发出的DNS域名解析请求，首先到达的就是本地名称服务器。

11.3.5 DNS报文格式

与HTTP报文一样，DNS报文也分为请求报文和应答报文两类，但DNS的请求报文和应答报文的总体格式是一样的，只是一些参数的取值不一样而已。DNS报文也分为两部分，一部分是报头部分，另一部分是数据部分。报头部分是固定的，有6个字段，共12字节大小，而数据部分由四大部分组成，且长度可变，如图11-11所示。下面主要介绍DNS报头部分以及数据部分中的“查询消息”和“应答消息”等内容。

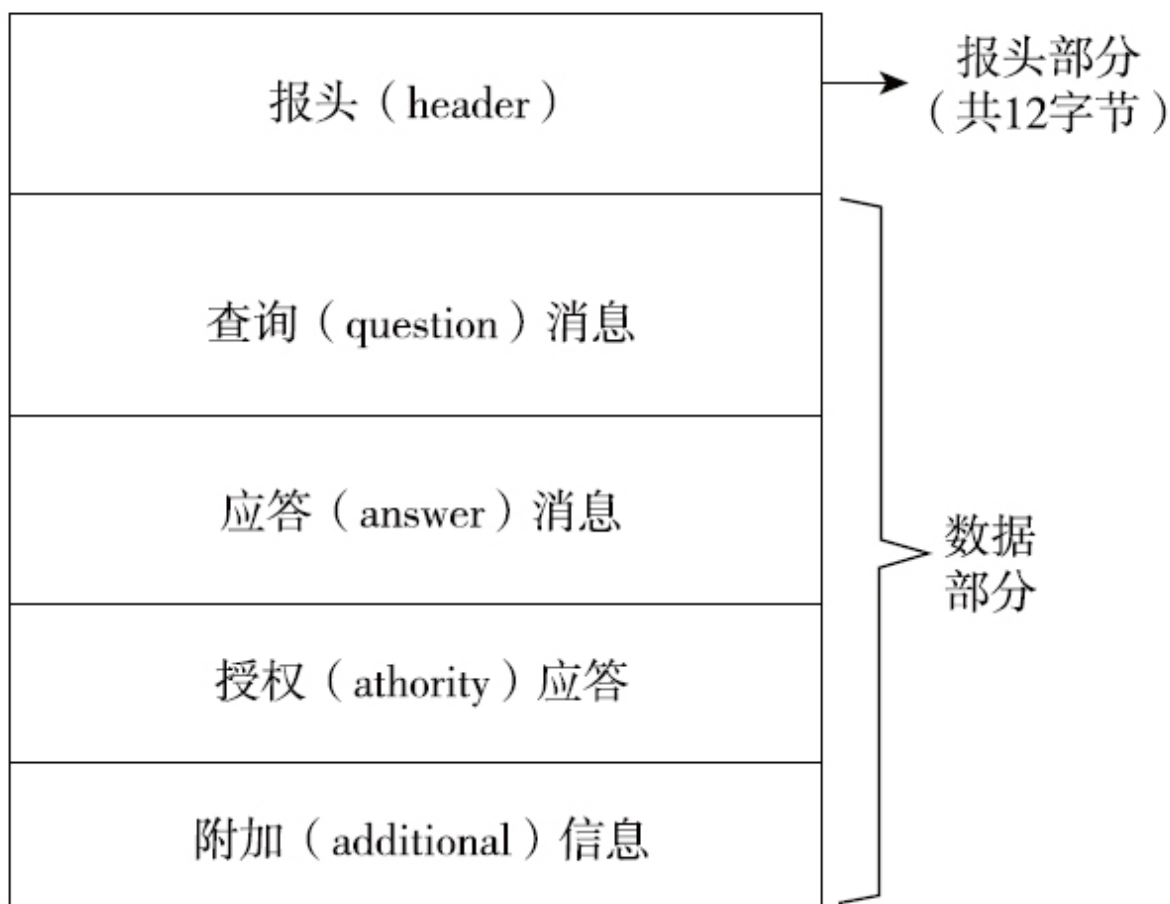


图 11-11 DNS报文基本格式

1.报头部分

DNS报头部分的格式如图11-12所示，下面是各字段的说明。

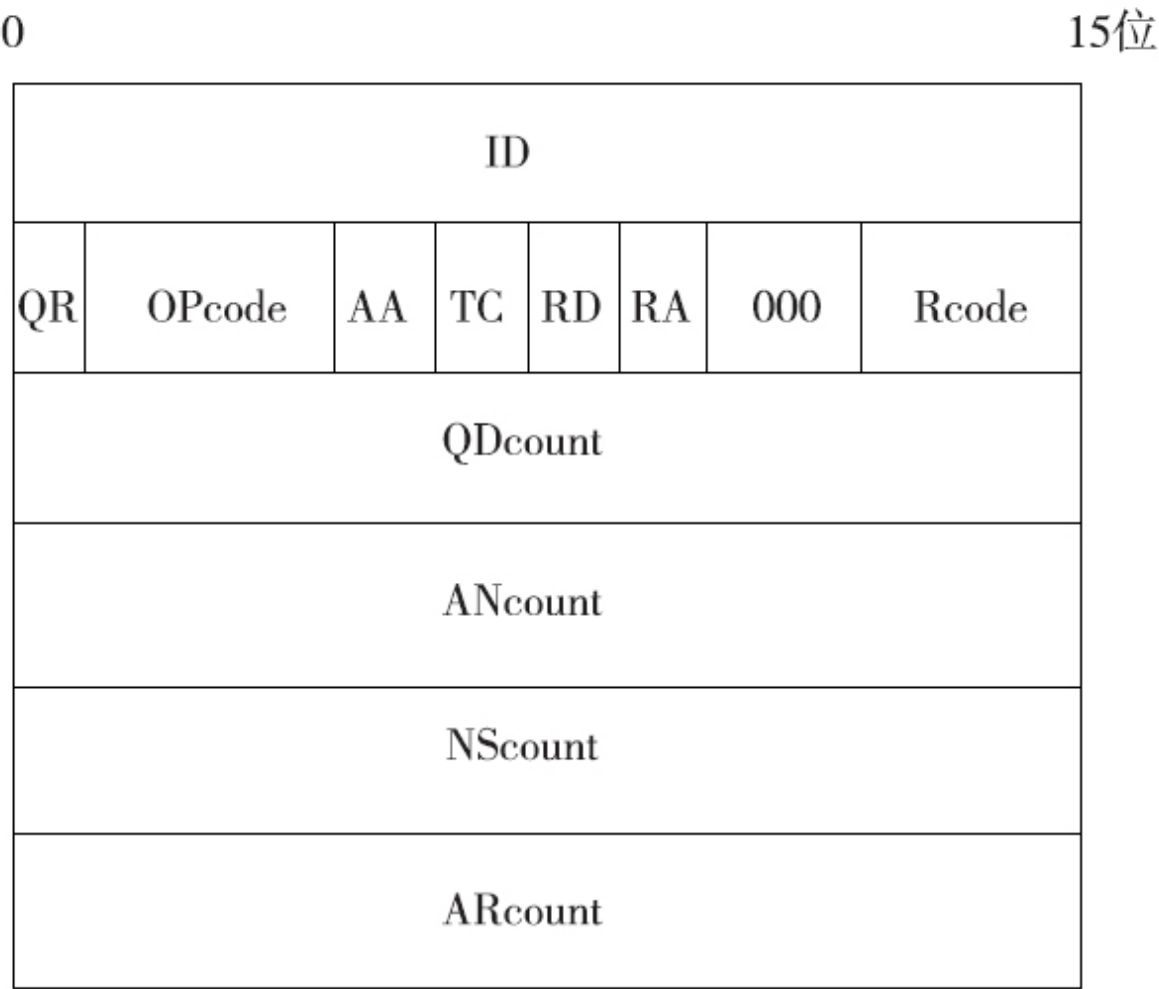


图 11-12 DNS报头部分的格式

□ID：这是由DNS查询程序指定的16位请求标识符，每次查询均会分配一个标识符，相当于查询报文序号。该标识符也会被随后的应

答报文所用，也就是应答报文中的标识符与其对应的请求报文中的标识符是一样的，这样就可以区分应答消息所对应的请求消息。

☐QR: 报文类型标志位，占1位，其中请求报文置0，应答报文置1。

☐OPcode: 操作码标志位，用于设置查询的种类，占4位，应答的时候会带相同值。置0时表示为标准查询（QUERY）；置1时表示为反向查询（IQUERY）；置2时表示为服务器状态查询（STATUS）；其他值保留，暂时未使用。

☐AA: 授权应答（Authoritative Answer）标志位，占1位，仅在应答报文中有意义。置1时表示在应答报文中所给出的名称服务器是所查询域名的权威名称服务器。

☐TC: 截断（TrunCation）标志位，占1位。置1时表示该报文超出最大报文长度，已被分段；置0时表示未被分段。

☐RD: 期望递归（Recursion Desired）标志位，占1位，在请求报文中设置。置1时表示建议域名服务器使用递归查询方法，应答的时候使用相同的值返回。

☐RA: 支持递归（Recursion Available）标志位，占1位，在应答报文中设置。置1时表示名称服务器支持递归查询，置0表示不支持。

后面紧随的是三个置0的位，用于保留使用。

□Rcode: 应答码 (Response code) 标志位，占4位，在应答报文中设置。置0时表示无错误，置1时表示报文格式有错误，名称服务器不接受请求的报文；置2时表示是由于服务器的原因而导致无法处理解析请求；置3时表示解析的域名不存在；置4时表示名称服务器不支持所请求的查询类型；置5时表示名称服务器由于设置的策略而拒绝给出应答。例如，服务器不希望对某些请求者给出应答，或者服务器不希望进行某些操作（如区域传输 (zone transfer)）。其他值保留，暂时未使用。

□QDcount: 16位整数，表示在报文后面“数据”部分“查询消息”字段中的问题条数。

□ANcount: 16位整数，表示在报文后面“数据”部分“应答消息”字段中的资源记录数。

□NScount: 16位整数，表示在报文后面“数据”部分“授权应答”字段中的名称服务器资源记录数。

□ARcount: 16位整数，表示在报文后面“数据”部分“附加信息”字段中的资源记录数。

2.请求消息格式

DNS请求报文中的具体请求消息是在如图11-11所示的报文格式中“查询消息”部分显示的，包括对应DNS域名查询所请求的问题。每条查询消息的格式如图11-13所示。

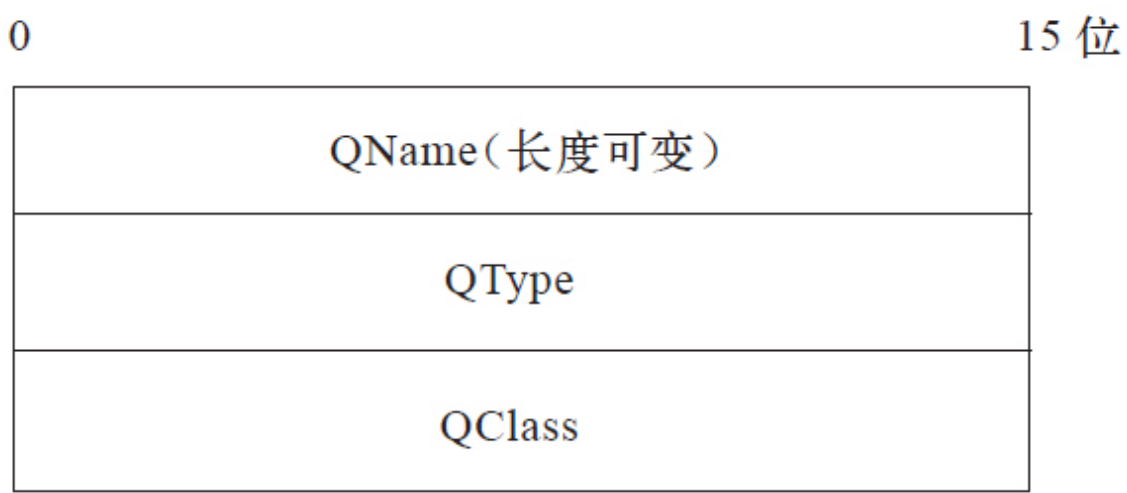


图 11-13 查询消息格式

1) QName: 表示所请求解析的域名，不过它显示的是一串ASCII编码，长度可变（注意，不是固定的16位）。在每个域名的开头以及中间的小圆点（.）部分均为一个十六进制数，表示下一节域名的ASCII字符数，后面是每节域名各个符号对应的ASCII字符（每个字母和符号均占1字节，即两位十六进制数字），在每个域名的最后均以一个字节的0（00）来表示。但要注意的是，这个字段的长度不是固定的，也可以为奇数个字节，不用填充。

例如，域名hi.baidu.com在请求消息中编码为02 68 69 05 62 61 69 64 75 03 63 6F 6D 00，其中“02”表示它的下一节域名中包括两个ASCII

字符，即“hi”，对应的ASCII字符为68 69；“05”表示它的下一节域名中包括5个ASCII字符，即“baidu”，对应的ASCII字符为62 61 69 64 75；“03”表示它的下一节域名中包括3个ASCII字符，即“com”，对应的ASCII字符为63 6F 6D；最后的“00”表示该域名结束。

2) QType: 表示查询的资源记录类型，占2字节，是本节后面将要介绍的资源记录中的TYPE（类型）字段的扩展。表11-7中的TYPE类型字段值均是Qtype字段合法的取值（有些通用的QType值可以和多条资源记录相匹配），另外，还可以有以下取值。

□AXFR: Authoritative Transfer（权威转换器），值为252，代表完整区域的域名解析请求。

□MAILB: Mail Box（邮箱），值为253，代表与邮箱相关的资源记录，如MB（邮箱）、MG（邮件组）或者MR（邮箱名更改）记录。

□MAILA: Mail Agent（邮件代理），值为254，代表邮件代理资源记录，如目前已基本不用的MX（邮件交换）记录。

□*: 值为255，代表所有资源记录的请求。

3) QClass: 表示查询类别，占2字节，是本节后面将要介绍的资源记录中的CLASS（类别）字段的扩展。表11-8中的CLASS字段值均

是QClass字段合法的取值。另外，新增了一个“*”查询类别，取值为255，代表所有分类。

3.资源记录格式

在DNS应答报文中，会在如图11-11所示的“应答消息”、“授权应答”和“附加信息”部分显示所应答的资源记录信息。这些资源记录消息格式是一样的，如图11-14所示。其中各字段说明如下。

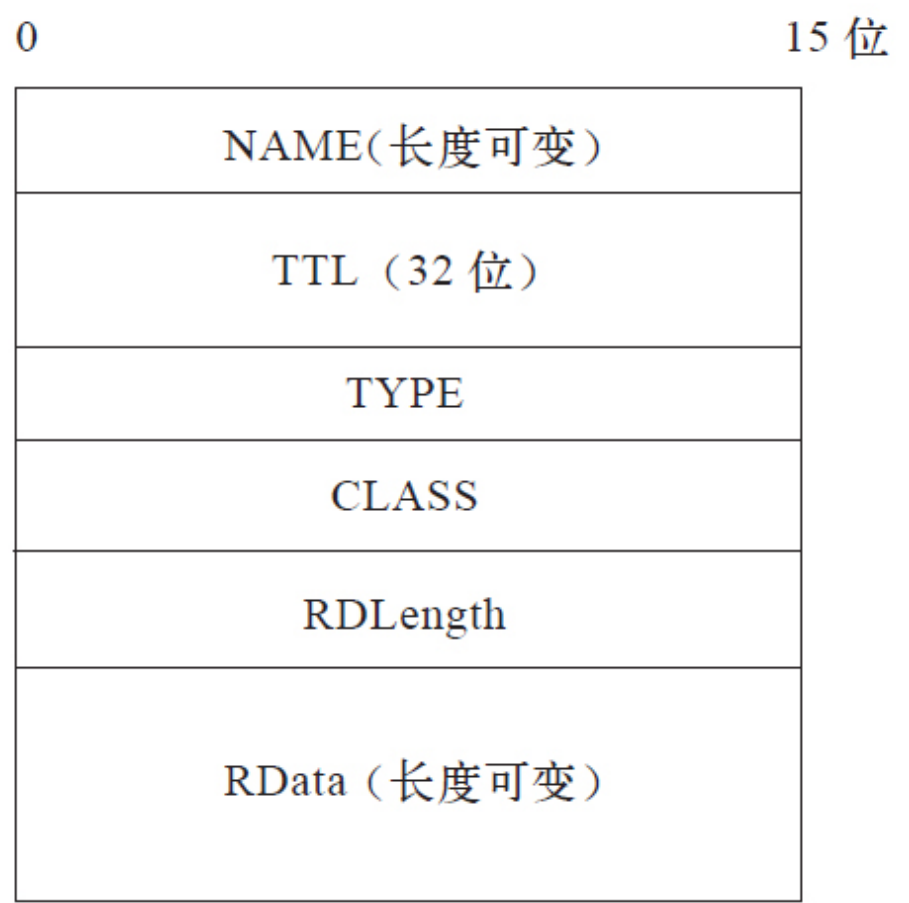


图 11-14 资源记录格式

□NAME: 表示资源记录对应的域名，与DNS请求报文QName字段所请求域名一致，该字段长度可变。

□TTL: 表示该资源记录可以缓存的时间（以秒为单位），一般用于当地址解析程序取出资源记录后决定保存及使用缓存数据的时间，占4字节，置0时表示只能被传输，不能被缓存。

□TYPE: 资源记录类型，占2字节。可用的资源记录类型及取值说明如表11-7所示。这里仅给出对应资源记录的类型，并不是具体的资源记录内容，具体的资源记录内容将在后面的RData字段中介绍。

表 11-7 TYPE 值类型及取值说明

资源记录类型	对应的字段值	说明
A	1	主机记录，给出一个主机 IP 地址。配置了多个 IP 地址的主机有对应数量的 A 记录
NS	2	名称服务器记录，给出一个权威名称服务器 IP 地址
MD	3	邮件目标记录，给出一个目的邮件地址。已过时，现用 MX 记录替代
MF	4	邮件转发器记录，给出一个邮件转发器。已过时，现用 MX 记录替代
CNAME	5	规范名记录，给出一个别名的规范名称
SOA	6	起始授权机构记录，给出一个区域的起始授权机构名称服务器 IP 地址
MB	7	邮箱记录，给出一个邮箱域名。仅用于实验
MG	8	邮件组记录，给出一个邮件组成员。仅用于实验
MR	9	邮件更名记录，给出一个邮件重命名后的域名。仅用于实验
NULL	10	空记录，给出一个空资源记录，相当于换行，主要是为了增加整个资源记录的可读性
WKS	11	熟知服务描述记录，给出一个熟知服务（像 HTTP、FTP、SMTP 这类常规服务）的描述
PTR	12	指针记录，给出一个 IP 地址的别名
HINFO	13	主机信息记录，给出主机操作系统和 CPU 信息
MINFO	14	邮箱信息记录，给出一个邮箱或邮件列表信息
MX	15	邮件交换记录，给出邮件交换的优先级，以及希望接受该域电子邮件的主机
TXT	16	文本记录，一个文本字符串

□CLASS: 资源记录数据的类别，占2字节，可用的数据类别及取值说明如表11-8所示，通常都是IN（即Internet）类别。这里仅给出对应资源记录数据（也就是RData字段中的内容）的类别，并不是具体的资源记录内容，具体的资源记录内容将在后面的RData字段中介绍。

表 11-8 CLASS 值类别及取值说明

资源记录数据类别	值	说明
IN	1	给出所请求解析域名对应的 IP 地址，是默认的资源记录数据类别
CS	2	CSNET（Computer Science Network，计算机科学网络）类别，目前基本不用
CH	3	Chaos 类别，由以前的 Symbolics Lisp 机器使用，已经被废弃
HS	4	Hesiod 类别，用来查询用户主目录，现在也不用了

□RDLength: 表示RData（资源数据）字段的长度（以字节为单位），占2字节。

□RData: 表示具体的相关资源记录，长度可变。其取值与TYPE、CLASS字段值有关。例如，如果TYPE值为A、CLASS值为IN，那么RData就是一个4字节的IP地址。

11.3.6 DNS数据传输方式

DNS服务同样也是C/S工作模式，分为DNS服务器和DNS客户端。其中，DNS客户端以DNS请求报文向DNS服务器发出域名解析请求，DNS服务器以DNS应答报文对客户端的DNS请求做出应答。

DNS既可以报文方式通过TCP 53号端口进行报文分组传输，也可以数据报方式通过UDP 53号端口进行数据报传输。至于何时使用TCP、何时使用UDP进行数据传输，需要从以下几个方面来考虑：

□当DNS数据（包括报头和数据部分）大于512字节时，只能采用TCP进行数据传输，因为UDP中传输的数据最长不能超过512字节。至于是否会分段，取决于具体网络中数据链路层的MTU值。如果被分段，则在DNS报头的TC标志位置1。

□在区域传输过程中，也就是在从主DNS服务器向辅助DNS服务器传输数据时必须使用TCP，因为这样传输更可靠。

□当DNS数据小于512字节时（这类情况比较少）是使用TCP，还是UDP，取决于DNS解析器，但请求和应答均使用相同的协议，不会出现请求时使用的是TCP，而在应答时使用的是UDP的情况。一般递归解析时使用UDP。

DNS域名解析其实很多是查找对应域名权威名称服务器，除非在本地名称服务器缓存中已有对应域名的记录。但事实上，名称服务器上的缓存容量和可以缓存的时间都有限，因此，绝大多数情况下是需要向权威名称服务器来求助解析的。

DNS有两种名称解析方式：一种称为“递归解析”（Recursive Solution, RS），另一种称为“迭代解析”（Iterative Solution, IS），也称“反复解析”。它们有不同的查询和应答流程：递归解析是一种代理查询方式，如果本地名称服务器不能解析，则后面的查询请求全由本地名称服务器代理DNS客户端进行查询，最终由本地名称服务器返回最后的解析结果；而迭代解析中的查询请求全由DNS客户端自己根据每次查询从上级DNS服务器上得到的信息依次查询下级DNS服务器。这就好比你要找一家你从未去过的公司，如果是通过一个服务公司为你去查找，然后把结果告诉你，这就相当于递归查询，而如果你是自己一步步去找人问，然后根据每次问话得到的信息去找其他人再问，这就是迭代查询。下面将分别介绍这两种名称解析方式。

11.3.7 DNS递归解析原理

“递归解析”（或称“递归查询”，其实意思是一样的）是最常见的默认的解析方式。在这种解析方式中，如果客户端配置的本地名称服务器不能解析的话，则后面的查询全由本地名称服务器代替DNS客户端进行查询，直到本地名称服务器从权威名称服务器得到了正确的解析结果，然后由本地名称服务器告诉DNS客户端查询的结果。

1.DNS递归解析基本流程

在这个查询过程中，一直是以本地名称服务器为中心的，DNS客户端只是发出原始的域名查询请求报文，然后就一直处于等待状态，直到本地名称服务器发来了最终的查询结果。此时的本地名称服务器就相当于中介代理的作用。如果考虑了本地名称服务器的缓存技术（也就是在DNS服务器上对一定数量的以前查询记录保存一定时间，这样后面查询同样的域名信息时就可直接从缓存中调出来，以加速查询效率）的话，则递归解析的基本流程如下。

- 1) 客户端向本机配置的本地名称服务器（在此仅以首选DNS服务器为例进行介绍，所配置的其他备用DNS服务器的解析流程也完全一样）发出DNS域名查询请求。

2) 本地名称服务器收到请求后，先查询本地的缓存，如果有该域名的记录项，则本地名称服务器直接把查询的结果返回给客户端；如果本地缓存中没有该域名的记录，则本地名称服务器再以DNS客户端的角色发送与前面一样的DNS域名查询请求给根名称服务器。

3) 根名称服务器收到DNS请求后，把查询到的所请求的DNS域名中顶级域名所对应的顶级名称服务器地址返回给本地名称服务器。

4) 本地名称服务器根据根名称服务器返回的顶级名称服务器地址，向对应的顶级名称服务器发送与前面一样的DNS域名查询请求。

5) 对应的顶级名称服务器在收到DNS查询请求后，也是先查询自己的缓存，如果有所请求的DNS域名的记录项，则先把对应的记录项返回给本地名称服务器，然后再由本地名称服务器返回给DNS客户端，否则向本地名称服务器返回所请求的DNS域名中的二级域名所对应的二级名称服务器地址。

然后，本地名称服务器继续按照前面介绍的方法一次次地向三级、四级名称服务器查询，直到最终的对应域名所在区域的权威名称服务器返回最终的记录给本地名称服务器，再由本地名称服务器返回给DNS客户，同时本地名称服务器会缓存本次查询得到的记录项。

2.DNS递归解析示例

为了方便读者理解，下面举例进行说明。本示例中假设客户端想要访问自己并不识别的example.microsoft.com站点，并假设此客户端配置的本地名称服务器为dns.company.com（通常是以IP地址方式配置的），本地名称服务器上配置的根名称服务器是a.rootserver.net。整个递归解析过程如图11-15所示（其中的Q1~Q5表示发送DNS查询请求，A1~A5是DNS查询应答），具体描述如下。

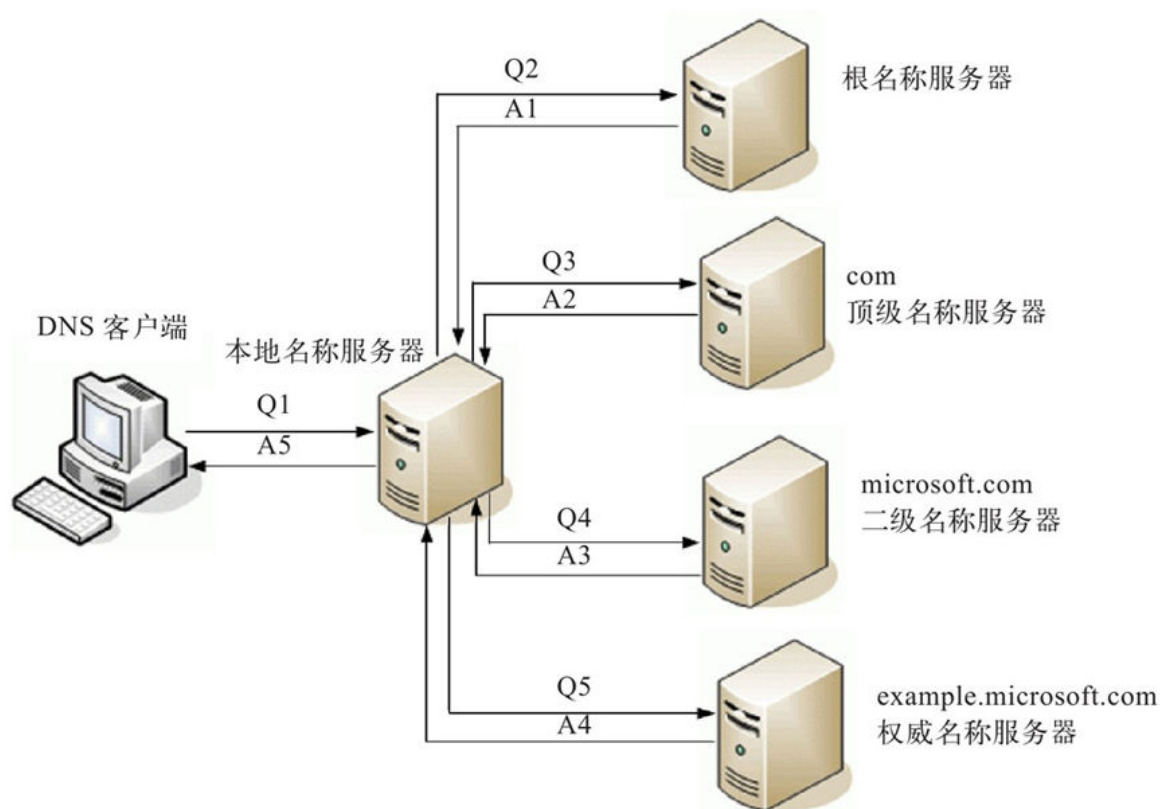


图 11-15 DNS递归解析示例

1) DNS客户端向所配置的本地名称服务器dns.company.com发出解析example.microsoft.com域名的DNS请求报文（图11-15中的Q1），相

当于对本地名称服务器说“请给我example.microsoft.com所对应的IP地址”。

2) 本地名称服务器收到请求后，先查询本地缓存，如果没有查到该域名的对应记录，则本地名称服务器向所配置的根名称服务器a.rootserver.net发出解析example.microsoft.com域名的DNS请求报文（图11-15中的Q2）。

3) 根名称服务器收到查询请求后，通过查询得到.com顶级域名所对应的顶级名称服务器，然后向本地名称服务器返回一条应答报文（图11-15中的A1），相当于说“我不知道example.microsoft.com域名对应的IP地址，但我现在告诉你.com域名对应的顶级名称服务器地址”。

4) 本地名称服务器在收到根名称服务器的DNS应答报文，并得到.com顶级域名所对应的顶级名称服务器地址后，再次向对应的顶级名称服务器发送一条请求解析example.microsoft.com域名的DNS请求报文（图11-15中的Q3）。

5) .com顶级名称服务器在收到DNS请求报文后，先查询自己的缓存，假设也没有该域名的记录项，则查询microsoft.com对应的二级名称服务器，然后也向本地名称服务器返回一条DNS应答报文（图11-15中的A2），相当于说“我不知道example.microsoft.com域名对应的IP地址，但我现在告诉你microsoft.com域名对应的二级名称服务器地址”。

6) 本地名称服务器在收到.com顶级名称服务器的DNS应答报文, 并得到microsoft.com二级域名所对应的二级名称服务器地址后, 再次向对应的二级名称服务器发送一条请求解析example.microsoft.com域名的DNS请求报文(图11-15中的Q4)。

7) microsoft.com二级名称服务器在收到DNS请求报文后, 也先查询自己的缓存, 如果也没有该域名的记录项, 则查询example.microsoft.com对应的权威名称服务器(因为这个名称服务器已包括了完整域名example.microsoft.com所在区域), 然后也向本地名称服务器返回一条DNS应答报文(图11-15中的A3), 相当于说“我不知道example.microsoft.com域名对应的IP地址, 但我现在告诉你example.microsoft.com域名对应的权威名称服务器地址”。

8) 本地名称服务器在收到microsoft.com二级名称服务器的DNS应答报文, 并得到example.microsoft.com三级域名对应的权威名称服务器地址后, 再次向对应的权威名称服务器发送一条请求解析example.microsoft.com域名的DNS请求报文(图11-15中的Q5)。

9) 权威名称服务器在收到DNS请求后, 在它的DNS区域数据库中查找, 最终会得出example.microsoft.com域名对应的IP地址, 然后向本地名称服务器返回一条DNS应答报文(图11-15中的A4), 相当于说“example.microsoft.com域名的IP地址为xxx.xxx.xxx.xxx”。

10) 本地名称服务器在收到权威名称服务器的应答报文后，向DNS客户端返回一条DNS应答报文（图11-15中的A5），告诉DNS客户端example.microsoft.com域名的IP地址。这样DNS客户端就可以正常访问这个网站了。

如果在步骤9) 中的对应域名的权威名称服务器中都找不到对应的域名记录，则会向本地名称服务器返回一条查询失败的DNS应答报文，这条报文最终也会由本地名称服务器返回给DNS客户端。当然，如果这个权威名称服务器上配置了指向其他名称服务器的转发器，则权威名称服务器还会在转发器指向的名称服务器上进一步查询。另外，如果DNS客户端上配置了多个DNS服务器，则还会继续向其他DNS服务器查询。

11.3.8 DNS迭代解析原理

在上面介绍的DNS递归解析中，当所配置的本地名称服务器解析不了时，后面的查询工作是由本地名称服务器替代DNS客户端进行的（以“本地名称服务器”为中心），只需要本地名称服务器向DNS客户端返回最终的查询结果即可。而本节介绍的DNS迭代解析（或者叫“迭代查询”）的所有查询工作全部是由DNS客户端自己完成的（以“DNS客户端”自己为中心）。在下列条件之一满足时就会采用迭代解析方式：

□在查询本地名称服务器时，如果客户端的请求报文中没有申请使用递归查询，那么在DNS请求报文头部的RD字段没有置1。相当于说，“你都没有主动要求我为你进行递归查询，我当然不会为你工作了”。

□客户端在DNS请求报文中申请使用的是递归查询（也就是RD字段置1了），但在所配置的本地名称服务器上禁用递归查询的（DNS服务器一般默认支持递归查询），即在DNS应答报文头部的RA字段置0。

1.迭代解析的基本流程

使用迭代解析方式时，如果它所配置的主名称服务器（如Windows操作系统中的“首选DNS服务器”）不能解析的话，那么客户端还会继续向所配置的其他名称服务器（如Windows操作系统中的“备用DNS服务器”）查询。迭代解析的基本流程如下。

- 1) 客户端向本机配置的本地名称服务器（在此仅以首选DNS服务器为例进行介绍，其他备用DNS服务器的解析流程也完全一样）发出DNS域名查询请求。

- 2) 本地名称服务器收到请求后，先查询本地的缓存，如果有该域名的记录项，则本地名称服务器直接把查询的结果返回给客户端；如果本地缓存中没有该域名的记录，则向DNS客户端返回一条DNS应答报文，报文中会给出一些参考信息，如本地名称服务器上的根名称服务器地址等。

- 3) DNS客户端在收到本地名称服务器的应答报文后，会根据其中的根名称服务器地址信息，向对应的根名称服务器再次发出与前面一样的DNS查询请求报文。

- 4) 根名称服务器在收到DNS查询请求报文后，通过查询自己的DNS数据库得到请求DNS域名中顶级域名所对应的顶级名称服务器信息，然后以一条DNS应答报文返回给DNS客户端。

5) DNS客户端根据来自根名称服务器应答报文中的对应顶级名称服务器地址信息，向该顶级名称服务器发出与前面一样的DNS查询请求报文。

6) 顶级名称服务器在收到DNS查询请求后，先查询自己的缓存，如果有所请求的DNS域名的记录项，则把对应的记录项返回给DNS客户端，否则通过查询后把对应域名中二级域名所对应的二级名称服务器地址信息以一条DNS应答报文返回给DNS客户端。

然后，DNS客户端继续按照前面介绍的方法一次次地向三级、四级名称服务器查询，直到最终的权威名称服务器返回最终的记录。

2.DNS迭代解析示例

为了方便读者理解，举例说明。本示例与11.3.7节介绍的DNS递归解析一样，即假设客户端想要访问自己并不识别的example.microsoft.com站点，并假设此客户端配置的本地名称服务器为dns.company.com（仅以一个本地名称服务器为例进行介绍），在该本地名称服务器上配置的根名称服务器是a.rootserver.net。整个迭代解析过程如图11-16所示（其中的Q1~Q5表示发送DNS查询请求，A1~A5是DNS查询请求的应答），具体描述如下。

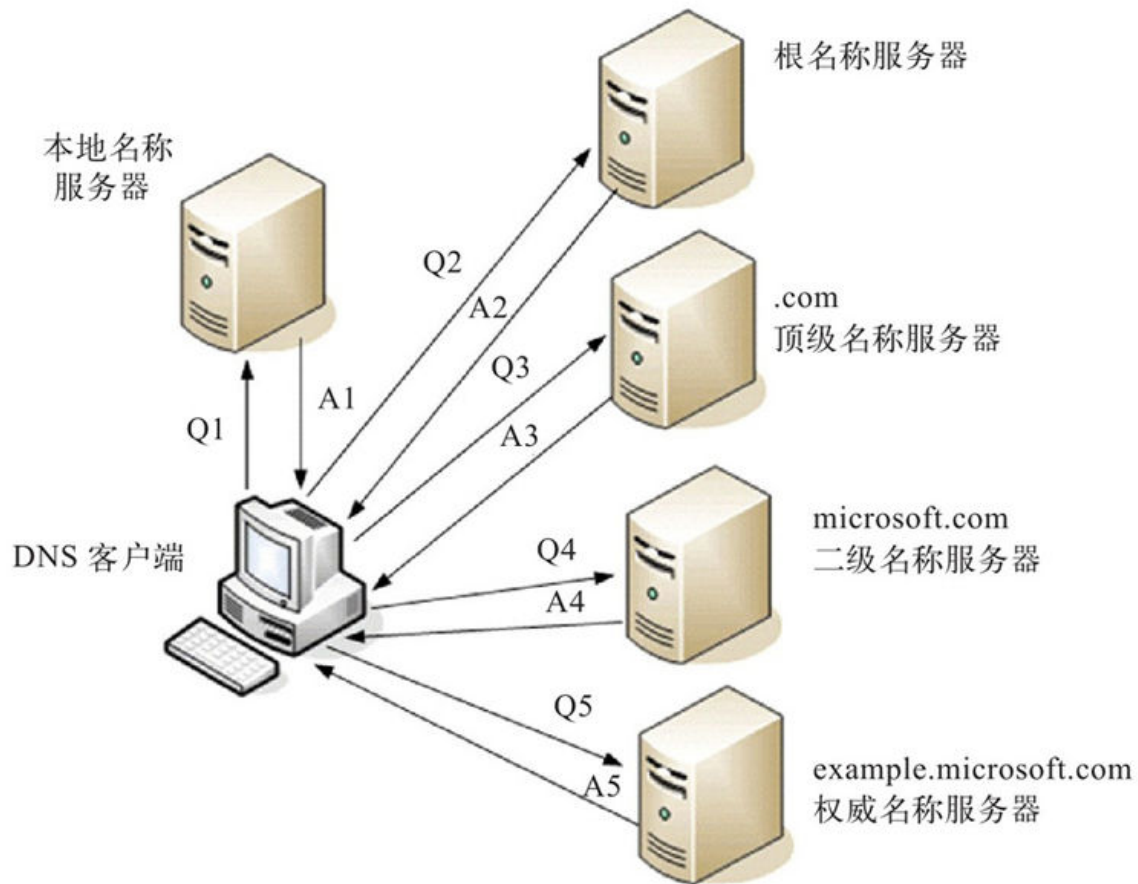


图 11-16 DNS迭代解析示例

1) DNS客户端向所配置的本地名称服务器`dns.company.com`发出解析`example.microsoft.com`域名的DNS请求报文（图11-16中的Q1）。

2) 本地名称服务器收到DNS客户端的DNS查询请求报文后，先查询本地缓存。如果没有查到该域名对应的记录，则本地名称服务器把所配置的根名称服务器`a.rootserver.net`地址信息以DNS应答报文返回给DNS客户端（图11-16中的A1）。

3) DNS客户端在收到本地名称服务器的DNS应答报文后，根据其中给出的根名称服务器地址信息，向对应的根名称服务器再次发送解析example.microsoft.com域名的DNS请求报文（图11-16中的Q2）。

4) 根名称服务器在收到DNS查询请求后，通过查询得到.com顶级域名对应的顶级名称服务器，然后把查询到的对应顶级域名信息以一条DNS应答报文返回给DNS客户端（图11-16中的A2）。

5) DNS客户端在收到根名称服务器的DNS应答报文，并得到.com顶级域名对应的顶级名称服务器地址后，再次向对应的顶级名称服务器发送一条解析example.microsoft.com域名的DNS请求报文（图11-16中的Q3）。

6) .com顶级名称服务器在收到DNS客户端的DNS查询请求报文后，先查询自己的缓存，假设也没有该域名的记录项，则查询microsoft.com对应的二级名称服务器，然后把查询到的对应二级域名信息以一条DNS应答报文返回给DNS客户端（图11-16中的A3）。

7) DNS客户端在收到.com顶级名称服务器的DNS应答报文，并得到microsoft.com二级域名对应的二级名称服务器地址后，再次向对应的二级名称服务器发送一条解析example.microsoft.com域名的DNS请求报文（图11-16中的Q4）。

8) microsoft.com二级名称服务器在收到DNS客户端的DNS查询请求报文后，也先查询自己的缓存，如果也没有该域名的记录项，则查询example.microsoft.com对应的权威名称服务器（因为这个名称服务器已包括了整个域名example.microsoft.com所在区域），然后把查询到的对应权威域名信息以一条DNS应答报文返回给DNS客户端（图11-16中的A5）。

9) DNS客户端在收到microsoft.com二级名称服务器的DNS应答报文，并得到example.microsoft.com三级域名对应的权威名称服务器地址后，再次向对应的权威名称服务器发送解析example.microsoft.com域名的DNS请求报文（图11-16中的Q5）。

10) 权威名称服务器在收到DNS客户端的DNS查询请求报文后，在它的DNS区域数据库中查找，最终会得出example.microsoft.com域名对应的IP地址，然后向DNS客户端返回一条DNS应答报文（图11-16中的A5）。这样DNS客户端就可以正常访问这个网站了。

如果在步骤10) 中的对应域名的权威名称服务器中都找不到对应的域名记录，则会向DNS客户端返回一条查询失败的DNS应答报文。当然，如果这个权威名称服务器上配置了指向其他名称服务器的转发器，则权威名称服务器还会在转发器指向的名称服务器上进一步查询。另外，如果DNS客户端上配置了多个DNS服务器，则还会继续向其他DNS服务器查询。

11.4 DHCP服务

DHCP（动态主机配置协议）是一种用于简化主机IP配置管理的服务。通过采用DHCP服务，可以使用DHCP服务器为网络上安装了DHCP服务客户端程序的客户端进行动态IP地址分配和其他相关设置，而不需要管理员对各个客户端进行一一配置，减轻了许多管理负担。

11.4.1 BOOTP和DHCP简介

DHCP是早期BOOTP（Bootstrap Protocol，自举协议）的增强版本。BOOTP是一个基于IP和UDP的协议，最早出现在UNIX系统中，负责UNIX终端的远程启动，后来在Windows无盘网络中也得到广泛使用。它可以让无盘站点从一个中心服务器上获得IP地址，为局域网中的无盘站点分配动态IP地址，并不需要每个用户去设置静态IP地址。

在使用BOOTP时，一般包括“自举协议服务端”和“自举协议客户端”两部分。在为无盘站点分配IP地址时，首先客户端网卡会以0.0.0.0地址向服务器发出IP地址分配请求的帧，其中包括站点网卡的MAC地址。BOOTP服务器在接收到这个请求帧后，根据这个帧中的MAC地址在自己的自举数据库（BOOTP database）中查找这个MAC的记录，

如果没有此**MAC**的记录，则不会应答这个请求；如果有，就向站点返回一条**FOUND**帧。**FOUND**帧中包含的主要信息有客户端的**IP**地址、服务器的**IP**地址、硬件类型、网关**IP**地址、站点**MAC**地址和启动映象文件名。站点根据**FOUND**帧中的信息获得分配的**IP**地址信息，然后通过**TFTP**服务器下载启动映象文件，并将此文件在站点内存模拟成磁盘，从这个模拟磁盘启动。

BOOTP有两个缺点：一是在为站点分配**IP**地址前，必须在服务器事先配置好对应站点的**MAC**地址，管理员的工作量比较大（主要是体现在事先必须收集各站点的**MAC**地址上）。二是**BOOTP**分配的**IP**地址是静态的，没有租约期的概念，即一个站点分配了一个**IP**地址后，可以永久使用这个**IP**地址，直到站点重启或关机。这样就形成了一个一对一的静态关系，不利于**IP**地址的有效使用。

DHCP不但是**BOOTP**的增强版本，而且是基于**IP**和**UDP**的。**DHCP**全面弥补了**BOOTP**的以上不足：在**DHCP**服务器端根本不用配置客户端的**MAC**地址，这样也就无须管理员去收集各客户端的**MAC**地址。另外，**DHCP**通过利用“租约”的技术可以大大提高**IP**地址的使用效率和安全性，不再是一对一的静态关系，而是一种动态分配关系。同时，在功能上，**DHCP**相对**BOOTP**有全面的增强，通过**DHCP**中继代理功能，一个**DHCP**服务器还可以为多个网段的客户端自动分配**IP**地址。

DHCP服务的目的是通过使用配置了DHCP服务的计算机或网络设备（如三层交换机、路由器，甚至防火墙）集中管理网络上使用的IP地址和其他相关配置，从而降低管理地址配置的复杂性。例如，在Windows 2000 Server/Windows Server 2003/Windows Server 2008以及各种Linux和UNIX服务器操作系统中都提供了DHCP服务，同时，各种三层交换机、路由器（包括宽带路由器）等也都全面支持了DHCP服务。注意，最新的DHCP草案标准为RFC 2131。

DHCP服务之所以能为DHCP客户端自动分配IP地址，其根本原因是在DHCP服务器中已准备好了用来为客户端分配IP地址的IP地址池。这个IP地址池就像装满了可用于分配的许多IP地址的池子一样，而且这些IP地址是属于一个网段的一部分或者全部的IP地址。有关DHCP的IP地址分配原理将在本章后续部分介绍。

11.4.2 DHCP服务的主要功能及应用环境

本节开始介绍一下DHCP服务的主要功能、使用DHCP服务的好处，以及DHCP服务的主要应用环境。

1.DHCP服务的主要功能

DHCP服务不仅提供简单的IP地址自动分配功能，还可以提供以下附加功能：

- 通过IP地址与MAC地址绑定功能实现静态IP地址的分配。
- 配置客户端的DNS服务器、WINS服务器（仅限Windows操作系统中的DHCP服务器）和默认网关。
- 利用IP地址排除功能，使已静态分配给其他主机（特别是各种服务器）的IP地址不再分配给其他DHCP客户端。
- 通过DHCP中继功能，一个DHCP服务器可以为多个网段（或VLAN）中的DHCP客户端分配不同地址池中的IP地址，进一步简化了网络中的IP地址配置工作。

2.使用DHCP服务的好处

在管理基于TCP/IP的网络中，使用DHCP服务有以下好处。

(1) 减少TCP/IP配置和管理的工作量

使用了DHCP服务后，安装了DHCP客户端程序（现在几乎所有操作系统都自带并安装了这一程序）的计算机可直接选择如图11-17所示的“自动获得IP地址”和“自动获得DNS服务器地址”选项，这样客户端就无须配置IP地址、子网掩码、网关、DNS服务器地址等TCP/IP信息。

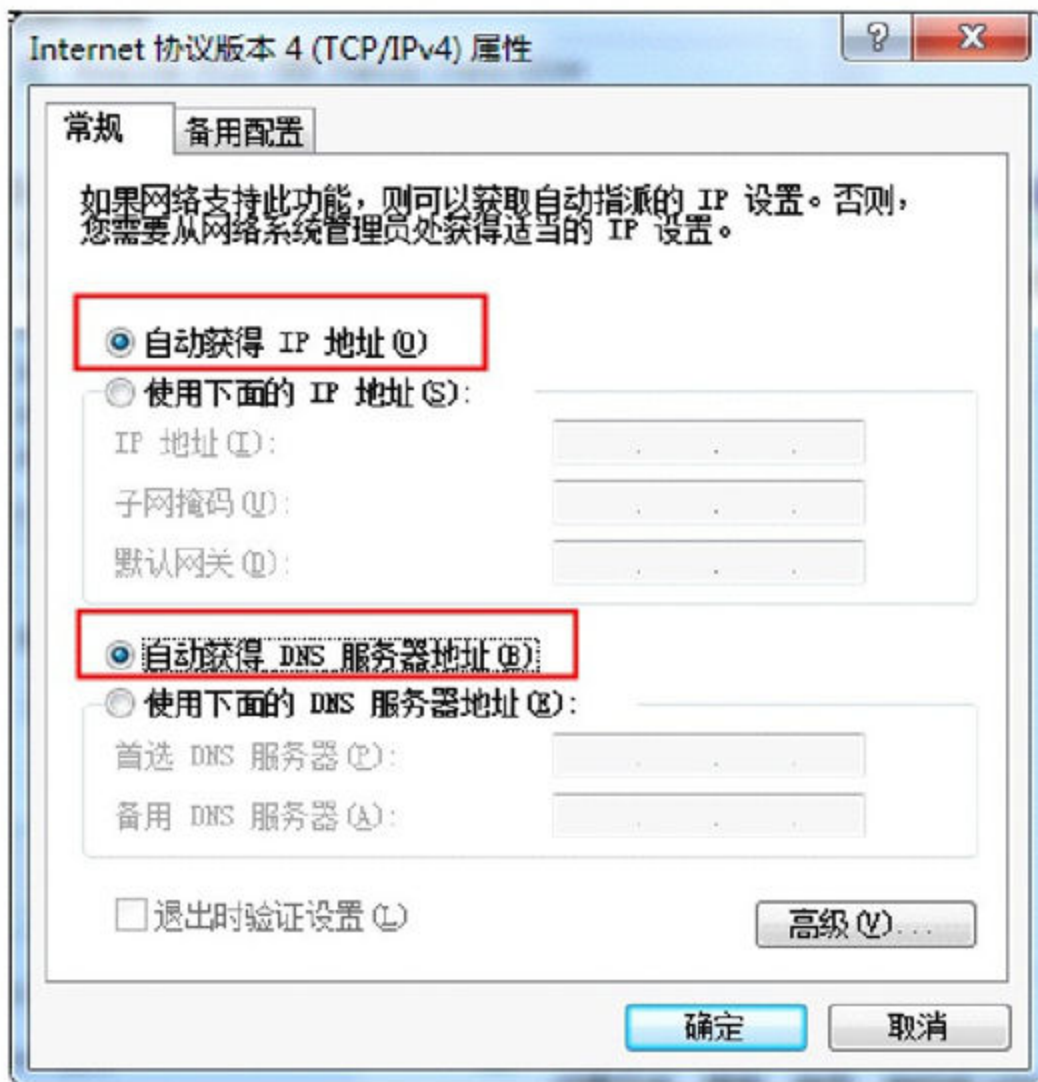


图 11-17 DHCP选项配置对话框

由此可见，DHCP服务可大大减轻管理员的工作负担（一般TCP/IP的配置是需要管理员权限的）。注意，由DHCP服务器自动分配的IP地址都有一个租约期，也就是自分配给某个客户端开始，该客户端只能在这个租约期内使用所分配的这个IP地址，但过期后可以续约，或者重新申请。

(2) 方便客户端移动

使用DHCP服务自动为客户端分配IP地址还有一个好处就是方便客户端的移动，这对于需要移动办公的人来说非常实用，这样客户端连接在哪里就可以从对应网段（或VLAN）配置的DHCP服务器上自动分配对应网段的IP地址。反过来，如果采用的是静态IP地址分配方式，则客户端每次移动到一个其他网段（或VLAN），管理员都需要重新设置一次TCP/IP信息。

(3) 配置更加可靠

采用DHCP服务自动分配IP地址信息可以有效地避免由于配置时的输入错误而引起的配置错误，还有助于防止在网络上配置新的计算机时因重用以前指派的IP地址而引起的地址冲突。

3.DHCP服务的主要应用环境

在下列场合，通常利用DHCP服务来完成IP地址的分配。

□网络规模较大，手工配置需要很大的工作量，并难以对整个网络进行集中管理。当然，像各种服务器、网络设备节点都是需要采用静态IP地址分配的，否则用户可能无法访问你的服务器，网络设备也无法进行正常的数据转发和路由。

□网络中主机数目大于该网络支持的IP地址数量，无法给每个主机分配一个固定的IP地址。例如，Internet接入服务提供商限制同时接入网络的用户数目，大量用户必须动态获得自己的IP地址。

□网络中只有少数主机需要固定的IP地址，大多数主机没有固定的IP地址需求。

11.4.3 DHCP报文及其格式

DHCP服务同样工作在C/S（客户端/服务器）模式中，但客户端与服务器进行报文传输时使用的UDP传输端口是不一样的，DHCP客户端使用UDP 68端口发送请求报文；DHCP服务器使用UDP 67端口发送应答报文。DHCP客户端向DHCP服务器发送的报文称为DHCP请求报文，而DHCP服务器向DHCP客户端发送的报文称为DHCP应答报文。

1.DHCP报文类型

整个DHCP服务一共有8种类型（主要是前面7种类型）的DHCP报文，分别为DHCP DISCOVER、DHCP OFFER、DHCP REQUEST、DHCP ACK、DHCP NAK、DHCP RELEASE、DHCP DECLINE、DHCP INFORM。上述报文类型的介绍如表11-9所示。

表 11-9 DHCP 报文类型

DHCP 报文类型	说 明
DHCP DISCOVER	因为 DHCP 客户端在请求 IP 地址时并不知道 DHCP 服务器的位置，因此 DHCP 客户端会在本地网络内以广播方式发送 DISCOVER 请求报文，以发现网络中的 DHCP 服务器。所有收到 DISCOVER 报文的 DHCP 服务器都会发送应答报文，DHCP 客户端据此可以知道网络中存在的 DHCP 服务器的位置
DHCP OFFER	DHCP 服务器收到 DISCOVER 报文后，就会在所配置的地址池中查找一个合适的 IP 地址，加上相应的租约期限和其他配置信息（如网关、DNS 服务器等），构造一个 OFFER 报文，发送给 DHCP 客户端，告知用户本服务器可以为其提供 IP 地址。但这个报文只是告诉 DHCP 客户端可以提供 IP 地址，最终还需要客户端通过 ARP 来检测该 IP 地址是否重复
DHCP REQUEST	因为 DHCP 客户端可能会收到很多 OFFER 请求报文，所以必须在这些应答中选择一个。通常是选择第一个 OFFER 应答报文的服务器作为自己的目标服务器，并向该服务器发送一个广播的 REQUEST 请求报文，通告选择的服务器，希望获得所分配的 IP 地址。另外，DHCP 客户端在成功获取 IP 地址后，在地址使用租期过去 1/2 时，也会向 DHCP 服务器发送单播 REQUEST 请求报文以请求续延租约，如果没有收到 ACK 报文，在租期过去 3/4 时，会再次发送广播的 REQUEST 请求报文以请求续延租约
DHCP ACK	当 DHCP 服务器收到 REQUEST 请求报文后，根据 REQUEST 报文中携带的用户 MAC 地址来查找有没有相应的租约记录，如果有，则发送 ACK 应答报文，通知用户可以使用分配的 IP 地址
DHCP NAK	如果 DHCP 服务器收到 REQUEST 请求报文后，没有发现有相应的租约记录或者由于某些原因无法正常分配 IP 地址，则向 DHCP 客户端发送 NAK 应答报文，通知用户无法分配合适的 IP 地址
DHCP RELEASE	当 DHCP 客户端不再需要使用分配 IP 地址时，就会主动向 DHCP 服务器发送 RELEASE 请求报文，告知服务器用户不再需要分配 IP 地址，请求 DHCP 服务器释放对应的 IP 地址
DHCP DECLINE	DHCP 客户端收到 DHCP 服务器 ACK 应答报文后，通过地址冲突检测机制发现服务器分配的地址冲突或者由于其他原因导致不能使用，则会向 DHCP 服务器发送 DECLINE 请求报文，通知服务器所分配的 IP 地址不可用，以期获得新的 IP 地址
DHCP INFORM	DHCP 客户端如果需要从 DHCP 服务器端获取更为详细的配置信息，则向 DHCP 服务器发送 INFORM 请求报文；DHCP 服务器在收到该报文后，将根据租约进行查找，当找到相应的配置信息后，向 DHCP 客户端发送 ACK 应答报文。注意，目前基本上不用了

2.DHCP报文格式

虽然DHCP服务的报文类型比较多，但每种报文的格式基本相同，只是某些字段取值可能不同。DHCP报文格式基于BOOTP的报文格式，如图11-18所示。下面是各字段的说明，其中具体取值参见11.4.4节给出的示例。

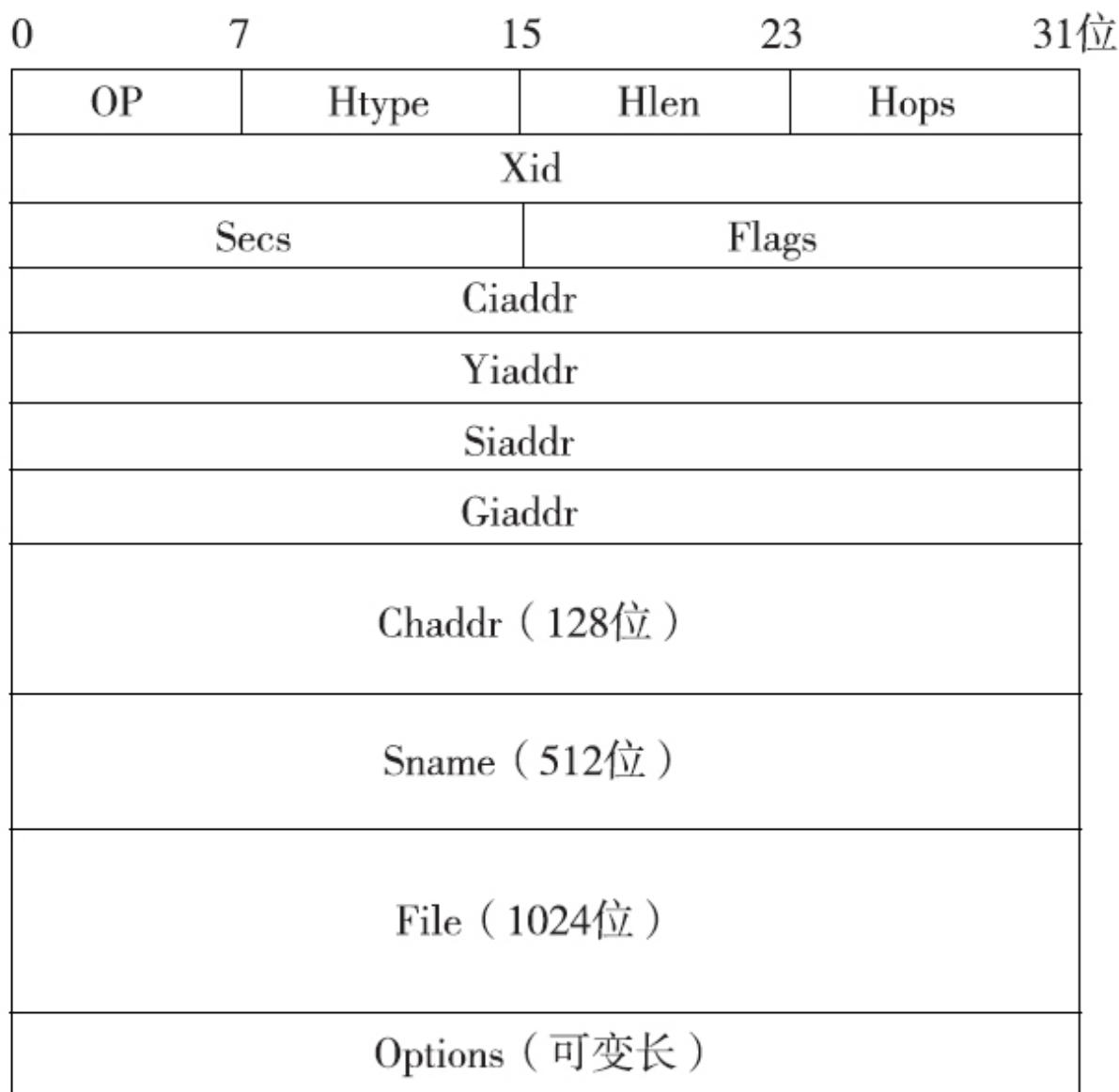


图 11-18 DHCP报文格式

□OP: Operation, 指定DHCP报文的操作类型, 占8位。请求报文置1, 应答报文置2。表11-9的报文类型中, DHCP DISCOVER、DHCP REQUEST、DHCP RELEASE、DHCP INFORM和DHCP DECLINE为请求报文, 而DHCP OFFER、DHCP ACK和DHCP NAK为应答报文。

□Htype、Hlen: 分别指定DHCP客户端的MAC地址类型及长度, 各占8位。MAC地址类型其实是指明网络类型, Htype字段置1时表示为最常见的以太网MAC地址类型。以太网MAC地址长度为6字节, 即对应Hlen字段值为6。

□Hops: DHCP报文经过的DHCP中继的数目, 占8位。DHCP请求报文每经过一个DHCP中继, 该字段就会增加1。没有经过DHCP中继时值为0。

□Xid: 客户端通过DHCP DISCOVER报文发起一次IP地址请求时所选择的随机数, 相当于请求标识, 占32位。用来标识一次IP地址请求过程。在一次请求中所有报文的Xid都是一样的。

□Secs: 表示DHCP客户端从获取到IP地址或者续约过程开始到现在所消耗的时间, 以秒为单位, 占16位。在没有获得IP地址前该字段始终为0。

□Flags: 标志位, 占16位, 第一位为广播应答标识位, 用来标识DHCP服务器应答报文是采用单播还是广播发送, 置0时表示采用单播发送方式, 置1时表示采用广播发送方式。其余位保留。

注意 在客户端正式分配了IP地址之前的第一次IP地址请求过程中, 所有DHCP报文都是以广播方式发送的, 包括客户端发送的DHCP DISCOVER和DHCP REQUEST报文, 以及DHCP服务器发送的DHCP

OFFER、DHCP ACK和DHCP NAK报文。当然，如果是由DHCP中继器转发的报文，则都是以单播方式发送的，具体内容在11.4.6节介绍。另外，IP地址续约、IP地址释放的相关报文都是采用单播方式发送的，具体内容在11.4.5节介绍。

□Ciaddr: 指示DHCP客户端的IP地址，占32位（4字节）。仅在DHCP服务器发送的ACK报文中显示，在其他报文中均显示0，因为在得到DHCP服务器确认前，DHCP客户端还没有分配到IP地址。

□Yiaddr: 指示DHCP服务器分配给客户端的IP地址，占32位（4字节）。仅在DHCP服务器发送的OFFER和ACK报文中显示，其他报文中显示为0。

□Siaddr: 指示下一个为DHCP客户端分配IP地址等信息的DHCP服务器IP地址，占32位（4字节）。仅在DHCP OFFER、DHCP ACK报文中显示，其他报文中显示为0。

□Giaddr: 指示DHCP客户端发出请求报文后经过的第一个DHCP中继的IP地址，占32位（4字节）。如果没有经过DHCP中继，则显示为0。

□Chaddr: 指示DHCP客户端的MAC地址，占128位（16字节）。在每个报文中都会显示对应DHCP客户端的MAC地址。

□Sname: 指示为DHCP客户端分配IP地址的DHCP服务器名称（DNS域名格式），占512位（64字节）。在OFFER和ACK报文中显示发送报文的DHCP服务器名称，其他报文显示为0。

□File: 指示DHCP服务器为DHCP客户端指定的启动配置文件名称及路径信息，占1024位（128字节）。仅在DHCP OFFER报文中显示，其他报文中显示为空。

□Options: 可选项字段，长度可变。

此部分可选的选项包含报文类型（代码为53，占1字节）、有效租约期（代码为51，以秒为单位，占4字节）、续约时间（代码为58，占4字节）、子网掩码（代码为1，占4字节）、默认网关（代码为3，可以是一个路由器IP地址列表，长度可变，但必须是4字节的倍数）、DNS服务器（代码为6，可以是一个DNS服务器IP地址列表，长度可变，但必须是4字节的整数倍）、域名称（代码为15，主DNS服务器名称，长度可变）、WINS服务器（代码为44，可以是一个WINS服务器IP列表，长度可变，但必须是4字节的倍数）等配置信息。表11-19所示的DHCP报文类型的取值分别如下。

□DHCP DISCOVER: 1

□DHCP OFFER: 2

☐DHCP REQUEST: 3

☐DHCP DECLINE: 4

☐DHCP ACK: 5

☐DHCP NAK: 6

☐DHCP RELEASE: 7

11.4.4 DHCP服务的IP地址自动分配原理

DHCP在提供服务时，DHCP客户端是通过UDP 68号端口进行数据传输的，而DHCP服务器则是以UDP 67号端口进行数据传输的。DHCP服务不仅体现在为DHCP客户端提供IP地址自动分配的过程中，还体现在后面的IP地址续约和释放过程中。本节仅介绍DHCP客户端初次分配IP地址的过程。

在DHCP服务器为DHCP客户端初次提供IP地址自动分配的整个过程中，一共经过了以下4个阶段，同时利用了表11-9中的前4个报文：发现阶段（DHCP客户端在网络中广播发送DHCP DISCOVER请求报文，发现DHCP服务器，请求IP地址租约）、提供阶段（DHCP服务器通过DHCP OFFER报文向DHCP客户端提供IP地址预分配）、选择阶段（DHCP客户端通过DHCP REQUEST报文确认选择第一个DHCP服务器为它提供IP地址自动分配服务）和确认阶段（被选择的DHCP服务器通过DHCP ACK报文把在DHCP OFFER报文中准备的IP地址租约给对应DHCP客户端）。

在DHCP客户端获得一个IP地址后，可以发送一个ARP请求来避免由于DHCP服务器地址池重叠而引发的IP冲突。以上4个阶段如图11-19

所示，相当于DHCP客户端与DHCP服务器的4次握手过程，具体描述如下。

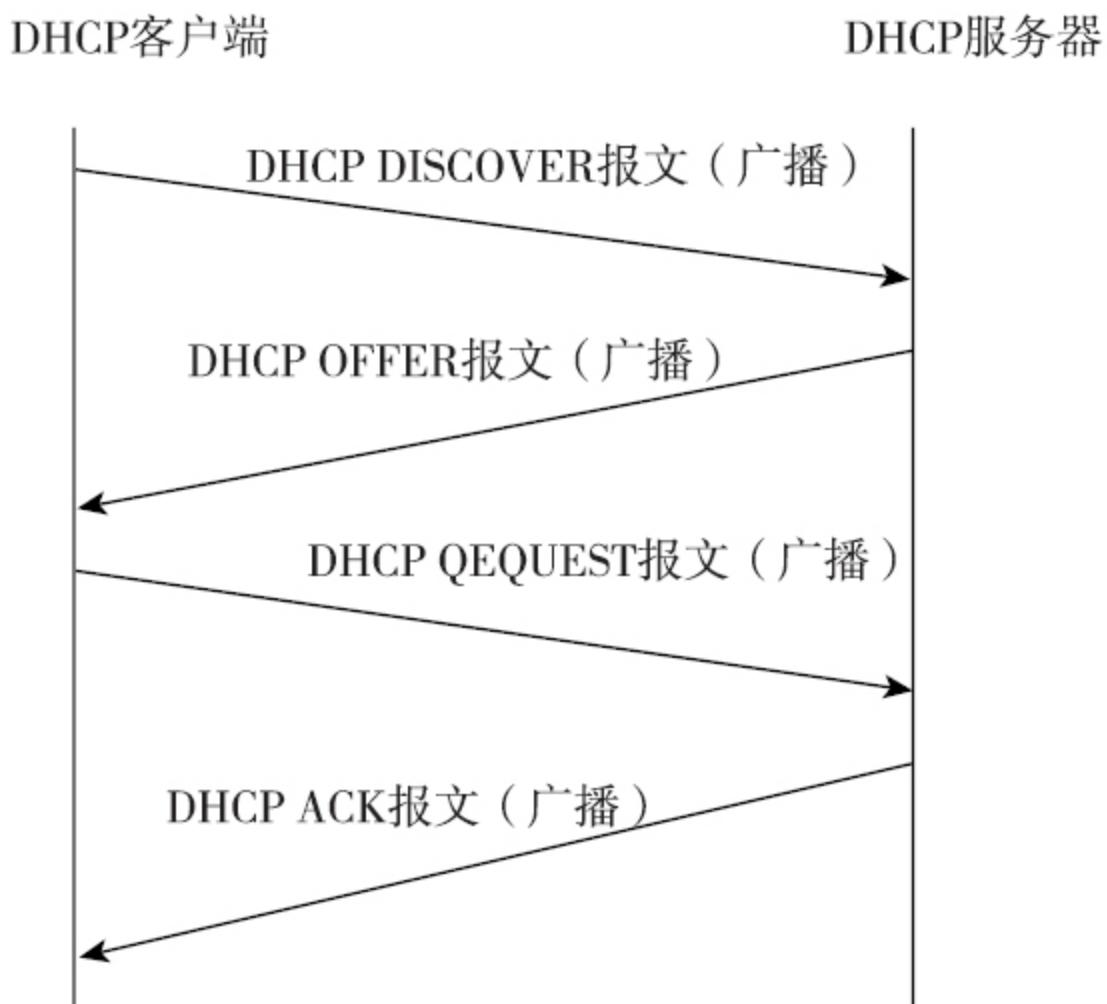


图 11-19 DHCP客户端从DHCP服务器获取IP地址的4个阶段

1) 发现阶段：即DHCP客户端获取网络中DHCP服务器信息的阶段。在客户端配置了DHCP客户端程序（如在Windows操作系统中进行了如图11-17所示的配置）并启动后，以广播方式发送DHCP DISCOVER报文寻找网络中的DHCP服务器。此广播报文采用传输层

的UDP 68号端口发送（封装的目的端口为UDP 68号端口），经过网络层IP封装后，源IP地址为0.0.0.0（因为此时还没有分配IP地址），目的IP地址为255.255.255.255（有限广播IP地址）。下面是一个DHCP DISCOVER报文封装的IP报头示例，可以看到Destination Address（目的地址）是255.255.255.255，而Source Address（源地址）是0.0.0.0。关于IP报头中的其他字段含义，可以参见7.3.4节相关内容。

```
IP:ID=0x0;Proto=UDP;Len:328
IP:Version=4(0x4)
IP:Header Length=20(0x14)
IP:Service Type=0(0x0)
IP:Precedence=Routine
IP:...0....=Normal Delay
IP:....0...=Normal Throughput
IP:.....0..=Normal Reliability
IP:Total Length=328(0x148)
IP:Identification=0(0x0)
IP:Flags Summary=0(0x0)
IP:.....0=Last fragment in datagram
IP:.....0.=May fragment datagram if necessary
IP:Fragment Offset=0(0x0)bytes
IP:Time to Live=128(0x80)
IP:Protocol=UDP-User Datagram !--- 使用UDP传输层协议
IP:Checksum=0x39A6
IP:Source Address=0.0.0.0 !--- 源IP地址为0.0.0.0
IP:Destination Address=255.255.255.255 !--- 目的IP地址为
255.255.255.255
IP>Data:Number of data bytes remaining=308(0x0134)
```

经验之谈 在上述DHCP DISCOVER报文中，IP报头中的目的地址是255.255.255.255。这个有限广播地址代表任意一个IPv4子网的广播地址，当然也是发送报文的主机所在的子网和DHCP服务器所在子

网的广播地址，但此时DHCP客户端并不知道DHCP服务器处于哪个子网。下面其他DHCP报文中的255.255.255.255地址的含义也是一样的。

对于IP报头中的源地址，由于当前DHCP客户端主机并未分配具体的IP地址，所以只能用具有任意代表功能的0.0.0.0地址来表示了。下面其他DHCP报文中指定的0.0.0.0地址的含义也是一样的。

此时，因为DHCP客户端没有分配到IP地址，也不知道DHCP服务器或DHCP中继的IP地址，所以在DHCP DISCOVER报文中Ciaddr（客户端IP地址）、Yiaddr（被分配的DHCP客户端IP地址）、Siaddr（下一个为DHCP客户端分配IP地址的DHCP服务器地址）、Giaddr

（DHCP中继IP地址）这4个字段均为0.0.0.0，如下所示。另外，从中可以看到，在CHADDR字段和DHCP选项的Client Identifier字段中都标识了DHCP客户端网卡MAC地址。

```
DHCP:Discover(xid=21274A1D)
DHCP:Op Code(op)=1(0x1)
DHCP:Hardware Type(htype)=1(0x1)10Mb Ethernet
DHCP:Hardware Address Length(hlen)=6(0x6)
DHCP:Hops(hops)=0(0x0)
DHCP:Transaction ID(xid)=556223005(0x21274A1D)
DHCP:Seconds(secs)=0(0x0)
DHCP:Flags(fags)=1(0x1) !--- 标志位置1，代表以广播方式发送
DHCP:1.....=Broadcast
DHCP:Client IP Address(ciaddr)=0.0.0.0
DHCP:Your IP Address(yiaddr)=0.0.0.0
DHCP:Server IP Address(siaddr)=0.0.0.0
DHCP:Relay IP Address(giaddr)=0.0.0.0
DHCP:Client Ethernet Address(chaddr)=08002B2ED85E
DHCP:Server Host Name(sname)=<Blank>
DHCP:Boot File Name(file)=<Blank>
DHCP:Magic Cookie=[OK]
```

```
DHCP:Option Field(options)
DHCP:DHCP Message Type=DHCP Discover !--- DHCP报文类型为DHCP
Discover
DHCP:Client-identifer=(Type:1)08 00 2b 2e d8 5e
DHCP:Host Name=JUMBO-WS !--- DHCP服务器主机名
DHCP:Parameter Request List=(Length:7)01 0f 03 2c 2e 2f 06
DHCP:End of this option field
```

2) 提供阶段：即DHCP服务器向DHCP客户端提供预分配IP地址的阶段。网络中的所有DHCP服务器接收到客户端的DHCP DISCOVER报文后，都会根据自己地址池中IP地址分配的优先次序选出一个IP地址，然后与其他参数一起通过传输层的UDP 67号端口，在DHCP OFFER报文中以广播方式发送给客户端（目的端口是DHCP客户端的UDP 68号端口）。客户端通过封装在帧中的目的MAC地址（也就是DHCP DISCOVER报文中的CHADDR字段值）的比对过程来确定是否接收该帧。但这样一来，理论上DHCP客户端可能会收到多个DHCP OFFER报文（当网络中存在多个DHCP服务器时），但DHCP客户端只接受第一个到来的DHCP OFFER报文。

DHCP OFFER报文经过IP封装后的源IP地址是DHCP服务器自己的IP地址，目的地址仍是255.255.255.255广播地址，使用的协议仍为UDP。下面是一个DHCP OFFER报文的IP报头示例。

```
IP:ID=0x3C30;Proto=UDP;Len:328
IP:Version=4(0x4)
IP:Header Length=20(0x14)
IP:Service Type=0(0x0)
IP:Precedence=Routine
IP:...0...=Normal Delay
IP:...0...=Normal Throughput
```

```
IP:.....0..=Normal Reliability
IP:Total Length=328(0x148)
IP:Identification=15408(0x3C30)
IP:Flags Summary=0(0x0)
IP:.....0=Last fragment in datagram
IP:.....0.=May fragment datagram if necessary
IP:Fragment Offset=0(0x0)bytes
IP:Time to Live=128(0x80)
IP:Protocol=UDP-User Datagram
IP:Checksum=0x2FA8
IP:Source Address=157.54.48.151
IP:Destination Address=255.255.255.255
IP:Data:Number of data bytes remaining=308(0x0134)
```

在DHCP OFFER报文中，Ciaddr字段值仍为0.0.0.0，因为客户端仍没有分配到IP地址；Yiaddr字段已有值了，这是DHCP服务器为该客户端预分配的IP地址；因为此时仍没有得到客户端确认，所以Siaddr字段值仍为0.0.0.0；因为没有经过DHCP中继服务器，所以Giaddr字段值仍为0.0.0.0。另外，在DHCP可选项部分，可以看到由服务器随IP地址一起发送的各种选项。在这种情况下，服务器发送的是子网掩码、默认网关（路由器）、租约时间、WINS服务器地址（NetBIOS名称服务）和NetBIOS节点类型。下面是一个DHCP OFFER报文示例。

```
DHCP:Offer(xid=21274A1D)
DHCP:Op Code(op)=2(0x2)
DHCP:Hardware Type(htype)=1(0x1)10Mb Ethernet
DHCP:Hardware Address Length(hlen)=6(0x6)
DHCP:Hops(hops)=0(0x0)
DHCP:Transaction ID(xid)=556223005(0x21274A1D)
DHCP:Seconds(secs)=0(0x0)
DHCP:Flags(flags)=1(0x1)
DHCP:1.....=Broadcast
DHCP:Client IP Address(ciaddr)=0.0.0.0
DHCP:Your IP Address(yiaddr)=157.54.50.5
DHCP:Server IP Address(siaddr)=0.0.0.0
DHCP:Relay IP Address(giaddr)=0.0.0.0
```

```
DHCP:Client Ethernet Address(chaddr)=08002B2ED85E
DHCP:Server Host Name(sname)=<Blank>
DHCP:Boot File Name(file)=<Blank>
DHCP:Magic Cookie=[OK]
DHCP:Option Field(options)
DHCP:DHCP Message Type=DHCP Offer !--- DHCP报文类型为DHCP OFFER
DHCP:Subnet Mask=255.255.240.0
!--- 所分配IP地址的子网掩码为255.255.240.0
DHCP:Renewal Time Value(T1)=8 Days,0:00:00
!--- 想要继续租约原来分配的IP地址, 则提出续约申请的期限为8天
DHCP:Rebinding Time Value(T2)=14 Days,0:00:00
!--- 如果上次申请续约失败, 再次申请绑定原来分配到的IP地址的期限为14天
DHCP:IP Address Lease Time=16 Days,0:00:00
!--- 租约期限为16天, 也就是DHCP客户端可使用此IP地址的最长时间为16天
DHCP:Server Identifier=157.54.48.151 !--- DHCP服务器的IP地址为
157.54.48.151
DHCP:Router=157.54.48.1 !--- 默认网关IP地址为157.54.48.1
DHCP:NetBIOS Name Service=157.54.16.154 !--- DNS服务器IP地址为
157.54.16.154
DHCP:NetBIOS Node Type=(Length:1)04
DHCP:End of this option field
```

3) 选择阶段: 即DHCP客户端选择IP地址的阶段。如果有多台DHCP服务器向该客户端发来DHCP OFFER报文, 则该客户端只接受第一个收到的DHCP OFFER报文, 然后以广播方式发送DHCP REQUEST报文。在该报文的“Requested Address”选项中, 包含DHCP服务器在DHCP OFFER报文中预分配的IP地址、对应的DHCP服务器IP地址等。这样也就相当于同时告诉其他DHCP服务器, 它们可以释放已提供的地址, 并将这些地址返回到可用地址池中。

在DHCP OFFER报文封装的IP头部中, 客户端的源地址仍是0.0.0.0, 数据包的目的地址仍是255.255.255.255。但在DHCP OFFER报

文中，Ciaddr、Yiaddr、Siaddr、Giaddr字段的地址均0.0.0.0。下面是一个DHCP OFFER报文头部和DHCP OFFER报文示例。

```
IP:ID=0x100;Proto=UDP;Len:328
IP:Version=4(0x4)
IP:Header Length=20(0x14)
IP:Service Type=0(0x0)
IP:Precedence=Routine
IP:...0....=Normal Delay
IP:....0...=Normal Throughput
IP:.....0..=Normal Reliability
IP:Total Length=328(0x148)
IP:Identification=256(0x100)
IP:Flags Summary=0(0x0)
IP:.....0=Last fragment in datagram
IP:.....0.=May fragment datagram if necessary
IP:Fragment Offset=0(0x0)bytes
IP:Time to Live=128(0x80)
IP:Protocol=UDP-User Datagram
IP:Checksum=0x38A6
IP:Source Address=0.0.0.0
IP:Destination Address=255.255.255.255
IP:Data:Number of data bytes remaining=308(0x0134)
DHCP:Request(xid=21274A1D)
DHCP:Op Code(op)=1(0x1)
DHCP:Hardware Type(htype)=1(0x1)10Mb Ethernet
DHCP:Hardware Address Length(hlen)=6(0x6)
DHCP:Hops(hops)=0(0x0)
DHCP:Transaction ID(xid)=556223005(0x21274A1D)
DHCP:Seconds(secs)=0(0x0)
DHCP:Flags(flags)=1(0x1)
DHCP:1.....=Broadcast
DHCP:Client IP Address(ciaddr)=0.0.0.0
DHCP:Your IP Address(yiaddr)=0.0.0.0
DHCP:Server IP Address(siaddr)=0.0.0.0
DHCP:Relay IP Address(giaddr)=0.0.0.0
DHCP:Client Ethernet Address(chaddr)=08002B2ED85E
DHCP:Server Host Name(sname)=<Blank>
DHCP:Boot File Name(file)=<Blank>
DHCP:Magic Cookie=[OK]
DHCP:Option Field(options)
DHCP:DHCP Message Type=DHCP Request
DHCP:Client-identifier=(Type:1)08 00 2b 2e d8 5e
DHCP:Requested Address=157.54.50.5
```

```
DHCP:Server Identifier=157.54.48.151
DHCP:Host Name=JUMBO-WS
DHCP:Parameter Request List=(Length:7)01 0f 03 2c 2e 2f 06
DHCP:End of this option field
```

4) 确认阶段：即DHCP服务器确认分配DHCP客户端IP地址的阶段。某个DHCP服务器在收到DHCP客户端发来的DHCP REQUEST报文后，只有DHCP客户端选择的服务器会进行如下操作：如果确认将地址分配给该客户端，则以广播方式返回DHCP ACK报文；否则返回DHCP NAK报文，表明地址不能分配给该客户端。

在DHCP服务器发送的DHCP ACK报文的IP头部，Source Address是DHCP服务器IP地址，Destination Address仍然是广播地址255.255.255.255。在DHCP ACK报文中的Yiaddr字段，包含要分配给客户端的IP地址，而ChADDR和DHCP:Client Identifier字段是发出请求的客户端中网卡的MAC地址。同时，在选项部分也会在DHCP OFFER报文中把所分配的IP地址的子网掩码、默认网关、DNS服务器、租约期、续约时间等信息加上。

```
IP:ID=0x3D30;Proto=UDP;Len:328
IP:Version=4(0x4)
IP:Header Length=20(0x14)
IP:Service Type=0(0x0)
IP:Precedence=Routine
IP:...0....=Normal Delay
IP:....0...=Normal Throughput
IP:.....0..=Normal Reliability
IP:Total Length=328(0x148)
IP:Identification=15664(0x3D30)
IP:Flags Summary=0(0x0)
IP:.....0=Last fragment in datagram
```

```
IP:.....0.=May fragment datagram if necessary
IP:Fragment Offset=0(0x0)bytes
IP:Time to Live=128(0x80)
IP:Protocol=UDP-User Datagram
IP:Checksum=0x2EA8
IP:Source Address=157.54.48.151
IP:Destination Address=255.255.255.255
IP:Data:Number of data bytes remaining=308(0x0134)
DHCP:ACK(xid=21274A1D)
DHCP:Op Code(op)=2(0x2)
DHCP:Hardware Type(htype)=1(0x1)10Mb Ethernet
DHCP:Hardware Address Length(hlen)=6(0x6)
DHCP:Hops(hops)=0(0x0)
DHCP:Transaction ID(xid)=556223005(0x21274A1D)
DHCP:Seconds(secs)=0(0x0)
DHCP:Flags(flags)=1(0x1)
DHCP:1.....=Broadcast
DHCP:Client IP Address(ciaddr)=0.0.0.0
DHCP:Your IP Address(yiaddr)=157.54.50.5
DHCP:Server IP Address(siaddr)=0.0.0.0
DHCP:Relay IP Address(giaddr)=0.0.0.0
DHCP:Client Ethernet Address(chaddr)=08002B2ED85E
DHCP:Server Host Name(sname)=<Blank>
DHCP:Boot File Name(file)=<Blank>
DHCP:Magic Cookie=[OK]
DHCP:Option Field(options)
DHCP:DHCP Message Type=DHCP ACK
DHCP:Renewal Time Value(T1)=8 Days,0:00:00
DHCP:Rebinding Time Value(T2)=14 Days,0:00:00
DHCP:IP Address Lease Time=16 Days,0:00:00
DHCP:Server Identifier=157.54.48.151
DHCP:Subnet Mask=255.255.240.0
DHCP:Router=157.54.48.1
DHCP:NetBIOS Name Service=157.54.16.154
DHCP:NetBIOS Node Type=(Length:1)04
DHCP:End of this option field
```

说明 客户端在收到服务器返回的DHCP-ACK确认报文后，会以广播的方式发送免费（gratuitous）ARP报文（该报文中，源IP地址和目标IP地址都是本机IP地址，源MAC地址是本机MAC地址，目的MAC地址是广播MAC地址），探测是否有主机使用服务器分配的IP地

址，如果在规定的时间内没有收到回应，客户端才使用此地址。否则，客户端会发送DHCP DECLINE报文给DHCP服务器，并重新申请IP地址。

如果网络中存在多个DHCP服务器，除DHCP客户端选中的服务器外，其他DHCP服务器中本次未分配出的IP地址仍可分配给其他客户端。

11.4.5 DHCP服务的IP地址租约更新原理

如果采用动态IP地址分配策略，则DHCP服务器分配给客户端的IP地址都是有一定的租约期限的，当租约期满后，DHCP服务器又会收回原来分配的这个IP地址。如果DHCP客户端希望继续使用该地址，则需要向DHCP服务器提出更新IP地址租约的申请，也就是前面所说的“续约”。IP地址租约更新，或者IP地址续约也就是更新服务器端对IP地址的租约信息，使其恢复为初始状态。申请续约的步骤如下：

1) 在DHCP客户端的IP地址租约期限达到1/2时，由DHCP客户端向为它分配IP地址的DHCP服务器以单播方式发送DHCP REQUEST请求报文，以期进行IP租约的更新。

2) 如果DHCP服务器同意续约，则DHCP服务器向客户端以单播方式返回DHCP ACK报文，通知DHCP客户端已经获得新IP租约，可以继续使用此IP地址；相反，如果DHCP服务器不同意续约，则DHCP服务器以单播方式返回DHCP NAK报文，通知DHCP客户端不能获得新的租约，此IP地址不可以再分配给该客户端。

3) 如果上面的续约申请失败，则DHCP客户端还会在租约期限达到7/8时，再次以广播方式发送DHCP REQUEST请求报文进行续约。DHCP服务器的处理方式同上，不再赘述。

如果第二次续约请求还是失败的，则原来租约的**IP**地址将被释放。

11.4.6 DHCP中继代理服务

在DHCP客户端初次从DHCP服务器获取IP地址的过程中，所有从DHCP客户端发出的请求报文和所有由DHCP服务器返回的应答报文都是以广播方式（目的地址为255.255.255.255）进行发送的，因此，DHCP服务只适用于DHCP客户端和DHCP服务器处于同一个子网（也就是DHCP服务器至少有一个端口是与DHCP客户端所在子网直接连接的）的情况，因为广播包是不能穿越子网的。

基于DHCP服务的以上限制，如果DHCP客户端与DHCP服务器之间存在路由器设备，不在同一子网，就不能直接通过这台DHCP服务器获取IP地址，即使DHCP服务器上已配置了对应的地址池。这也就意味着，如果想要让多个子网中的主机进行动态IP地址分配，就需要在网络中的所有子网中都设置一个DHCP服务器，这显然是很不合算的，也是没有必要的。

DHCP中继功能可以很好地解决DHCP服务的以上难题。通过DHCP中继代理服务，与DHCP服务器不在同一子网的DHCP客户端可以通过DHCP中继代理（通常由路由器或三层交换机设备担当，但需要开启DHCP中继功能）与位于其他网段的DHCP服务器通信，最终使DHCP客户端获取到从DHCP服务器上分配而来的IP地址。此时的DHCP中继代理就位于DHCP客户端和DHCP服务器之间，负责广播

DHCP报文的转发，如图11-20所示。显然，DHCP中继代理必须有一个端口连接了DHCP客户端所在子网，另一端又直接连接了DHCP服务器所在子网。当然，一个DHCP中继代理可以同时连接多个子网，作为多个DHCP客户端子网的中继代理。这样，多个子网上的DHCP客户端又可以使用同一个DHCP服务器来自动分配IP地址，既节省了成本，又便于集中管理。

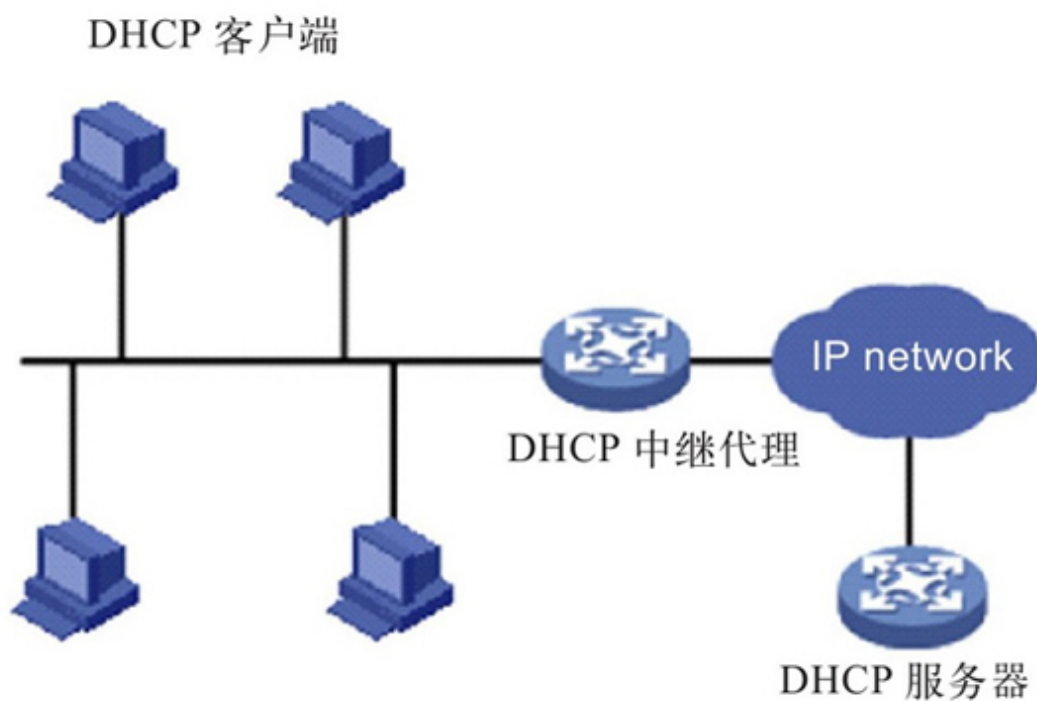


图 11-20 DHCP中继代理的典型应用示例

经验之谈 其实这里的DHCP中继代理相当于网关，DHCP广播报文都是通过这个网关来进行转发的。DHCP中继代理服务的典型应用环境就是一个DHCP服务器为所连接的交换机上的多个VLAN分配IP地址。这时因为DHCP服务器只能属于一个VLAN，所以要为多个VLAN

同时分配IP地址的话，就必须在这些VLAN与DHCP服务器之间设置一个DHCP中继代理，那就是需要在三层交换机上开启DHCP中继代理服务，让三层交换机充当DHCP中继代理角色。然后，在DHCP服务器上启用802选项（用于识别DHCP中继代理），并在每个地址池中为每个子网配置一个默认网关地址（各VLAN接口IP地址）即可。

1.DHCP中继代理简介

Option 82是DHCP报文中的中继代理信息选项（relay agent information option），该选项记录了DHCP客户端的位置信息。管理员可以利用该选项定位DHCP客户端，实现对客户端的安全和计费等控制。支持Option 82选项的DHCP服务器还可以根据该选项的信息制定IP地址和其他参数的分配策略，提供更加灵活的地址分配方式。

在整个Option 82选项中，最多又可以包含255个子选项，但至少要定义一个子选项。目前设备主要只支持两个子选项：sub-option 1

（Circuit ID，电路ID子选项）和sub-option 2（Remote ID，远程ID子选项）。由于RFC 3046对于Option 82的内容没有统一规定，因此不同厂商通常根据需要进行填充。目前，DHCP中继设备都支持Option 82子选项的扩展填充格式。

sub-option 1和sub-option 2两个子选项格式分别如图11-21和图11-22所示，其中固定字段（相当于子选项头）括号中的内容为该字段的固

定取值。sub-option 1子选项中的内容是接收到DHCP客户端请求报文的端口所属VLAN的编号（对应“VLAN ID”字段，占2字节）以及端口索引（端口索引的取值为端口物理编号减1，对应“Port Index”字段，占2字节）。sub-option 2子选项的内容是接收到DHCP客户端请求报文的DHCP中继设备的MAC地址（对应“MAC Address”字段，占6字节）。

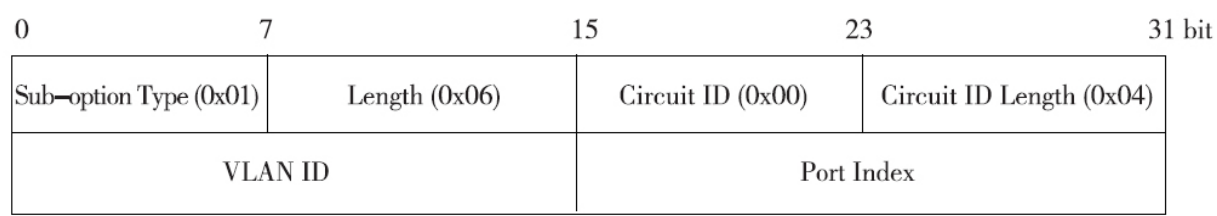


图 11-21 sub-option 1子选项格式及默认填充

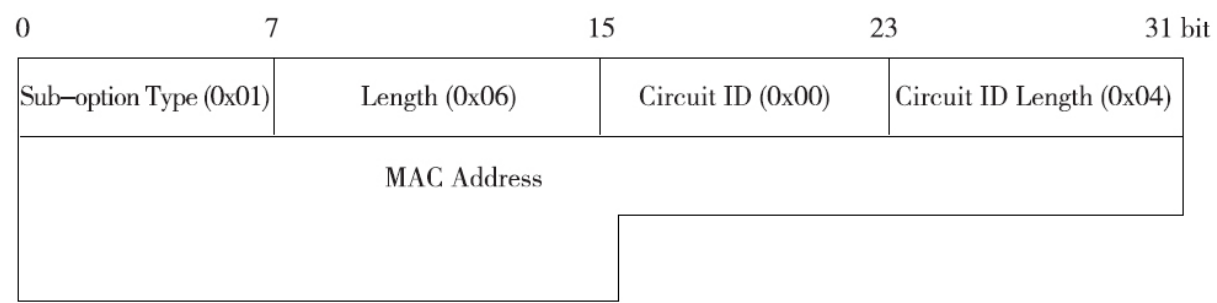


图 11-22 sub-option 2子选项格式及默认填充

2.通过DHCP中继代理动态分配IP地址的原理

DHCP客户端通过DHCP中继代理服务从DHCP服务器动态获取IP地址的过程与11.3.4节中介绍的不通过DHCP中继代理而直接从DHCP服务器动态获取IP地址的过程基本相同，都要经历发现、提供、选择和确认4个阶段。其中所用的到报文也对应是DHCP DISCOVER、DHCP

OFFER、DHCP REQUEST、DHCP ACK，只是在这里DHCP中继代理需要充当一个中介代理的角色，负责转发DHCP客户端与DHCP服务器之间交互的这些报文，基本流程如图11-23所示。

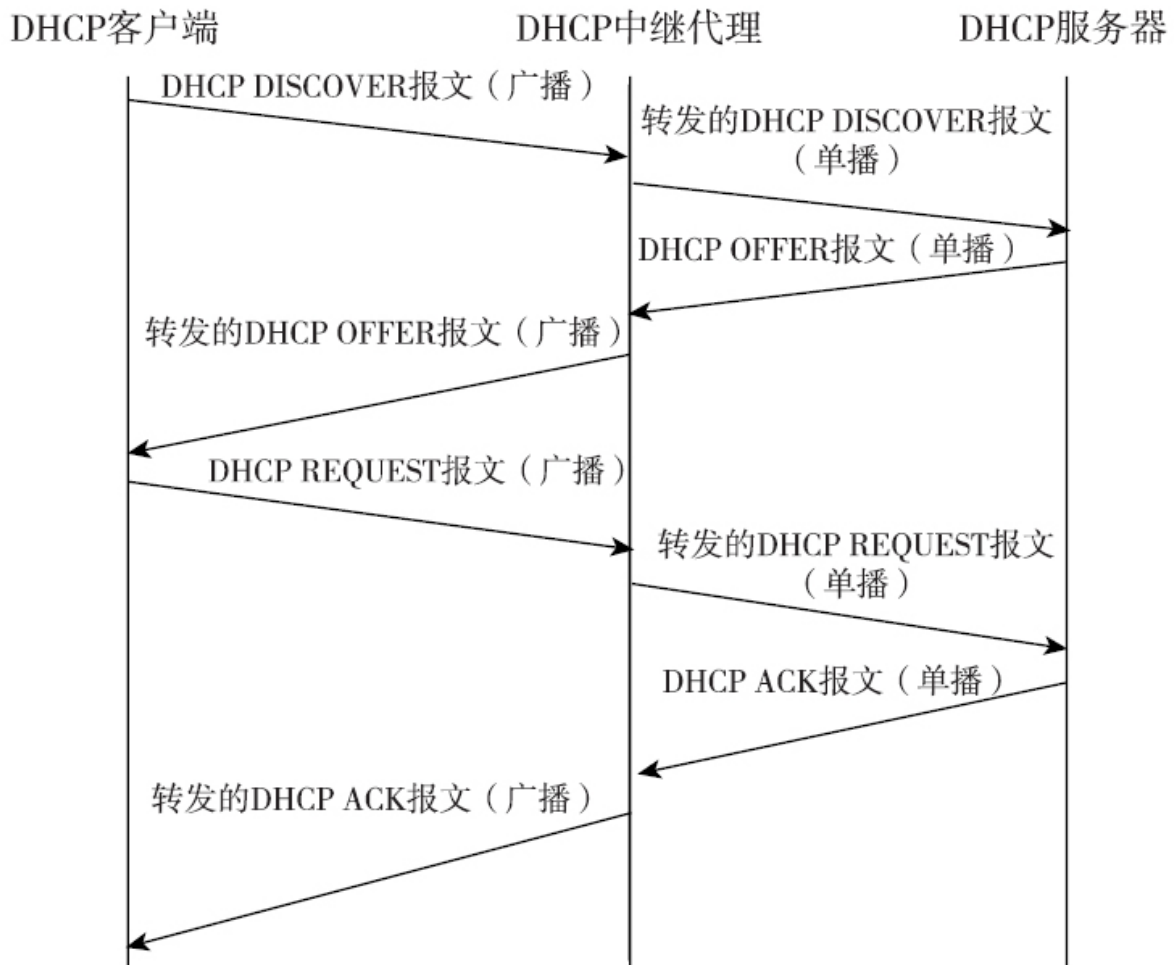


图 11-23 DHCP中继代理基本流程

说明 在如图11-23所示的流程中，DHCP客户端发送的请求报文有两种，分别为DHCP DISCOVER报文和DHCP REQUEST报文。由于不同厂商生产的DHCP服务器设备对请求报文的处理机制不同，有些设

备处理DHCP DISCOVER报文中的Option 82信息，而有些处理DHCP REQUEST报文中的Option 82信息，因此，DHCP中继设备将在这两种报文中都添加Option 82选项。

这里将只介绍DHCP中继支持Option 82时的工作机制，具体如下：

1) DHCP客户端以广播方式（因为不知道中继代理设备的IP地址）向本网段发送DHCP DISCOVER或DHCP REQUEST请求报文。此时，只有网络中的DHCP中继代理设备会接收该DHCP请求报文（假设本网段中没有DHCP服务器）。

2) DHCP中继代理设备在收到DHCP客户端发来的DHCP DISCOVER或DHCP REQUEST请求报文后，将检查报文中是否已有Option 82选项（主要是前面介绍的sub-option 1子选项和sub-option 2子选项）。

如果请求报文中已有Option 82，则DHCP中继代理设备会按照配置的策略对该报文进行处理（丢弃，或者用中继设备本身的Option 82选项替代报文中原有的Option 82选项，或者保持报文原有的Option 82选项）。同时，根据Option 82选项的sub-option 1子选项中的VLAN ID和Port Index字段配置，找到为对应网段分配的DHCP服务器地址，并将请求报文中的giaddr字段填充为DHCP中继代理设备的IP地址，然后将请

求报文以单播方式（因为在中继代理设备中已配置好了对应的DHCP服务器地址）转发给指定的DHCP服务器。

如果请求报文中没有Option 82选项，则DHCP中继设备将Option 82选项添加到报文中后，根据Option 82选项的sub-option 1子选项中的VLAN ID和Port Index字段配置，找到为对应网段分配的DHCP服务器地址，并将报文中的giaddr字段填充为DHCP中继代理设备的IP地址，然后同样根据中继代理设备中为对应网段配置的DHCP服务器地址以单播方式将请求报文转发给DHCP服务器。

3) DHCP服务器在收到由DHCP中继代理设备转发的DHCP DISCOVER或DHCP REQUEST请求报文后，根据转发的请求报文中的giaddr字段值所对应的中继代理设备IP地址，以及在Option 82选项的sub-option 2子选项中的中继代理MAC地址，以单播方式向DHCP中继代理返回对应的DHCP OFFER或者DHCP ACK应答报文。

4) DHCP中继设备在收到DHCP服务器的应答报文后，将剥离报文中的Option 82信息，然后以广播方式将带有DHCP配置信息的对应应答报文转发给DHCP客户端，完成对客户端的动态配置。

在以上过程中，4种DHCP报文中除Option 82选项和giaddr字段外，其他各字段值均与11.3.4节介绍的非中继方式动态分配IP地址过程中的4种报文所对应的字段值是一样的。

11.5 电子邮件服务

电子邮件服务（E-mail服务）类似于传统的邮件服务（如中国邮政集团、各家快递公司提供的信件、包裹邮寄服务），只是它是电子化的，不仅需要邮寄的文件要电子化（也就是计算机文件），而且邮寄的方式也要电子化（通过互联网邮寄）。电子邮件服务自20世纪90年代得到普及后，传统的邮件服务和电报服务都受到了很大冲击，因为电子邮件服务比传统邮件服务更快，可靠性更高，最重要的是免费的，只要有互联网接入的地方都可以随时收、发电子邮件。

11.5.1 电子邮件系统的基本结构

电子邮件服务也是一种基于C/S模式的服务，因此，它也分为电子邮件客户端和电子邮件服务器。但是要注意的是，其实在完整的电子邮件系统中，包括发件方和收件方都会包括一套客户端和服务系统。电子邮件不是直接从发送方传输到接收方的，而是通过双方的电子邮件服务器转发、保存。这时，各自的电子邮件服务器就相当于一个存储电子邮件的“仓库”和用于非即时电子邮件转发的中转站。这样做有两方面的好处：

□用户不必时刻在线等待电子邮件的到来，这个工作直接由自己选择的电子邮件服务器来完成，因为电子邮件服务器，特别是ISP提供的电子邮件服务器是每天24小时不关机的。电子邮件服务器可以每天24小时随时接收，并保存用户的电子邮件，用户想何时收邮件都可以。

□用户自己不必识别互联网上的各种各样的电子邮件域名，这个工作也全部交给由ISP提供的电子邮件服务器完成。这样可大大降低电子邮件用户主机的配置难度，只需安装一个电子邮件客户端软件，然后在客户端中配置好自己申请的ISP电子邮箱账户，以及对应的电子邮件服务器地址。

整个电子邮件系统包括“用户代理”（user agent）和“消息传输代理”（message transfer agent）两大部分。“用户代理”负责电子邮件用户书写、发送、接收和阅读电子邮件，是安装在用户端上的各种电子邮件客户端软件，是用户与电子邮件系统间交互的本地程序。“消息传输代理”负责电子邮件在发件方的客户端和收件方的服务器端之间，以及在发件方的服务器端与收件方的服务器端之间的传输，是安装在ISP或者本地网络中的各种电子邮件服务器软件。“消息传输代理”也称“守护进程”（daemon），是在服务器程序后台运行的应用服务进程。基本电子邮件系统结构如图11-24所示。

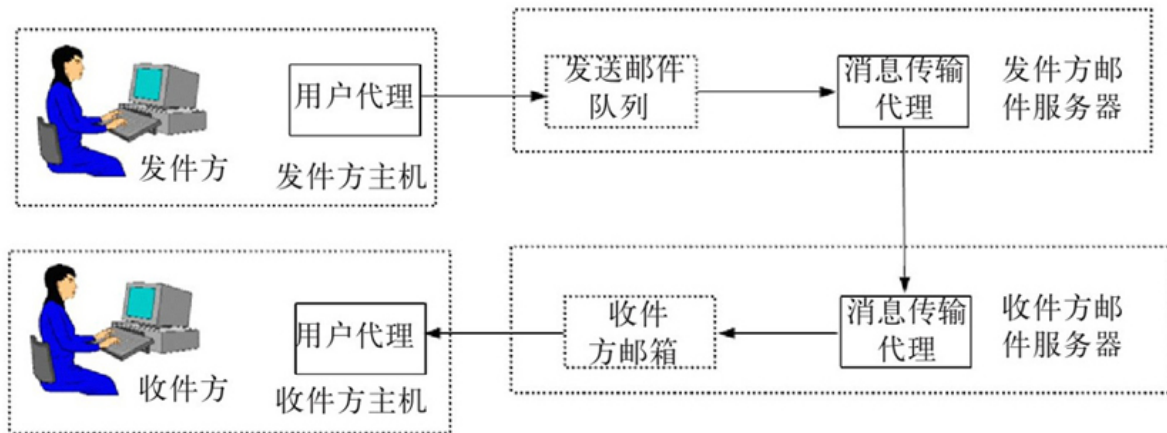


图 11-24 基本电子邮件系统结构

与其他计算机网络应用服务一样，电子邮件服务也是由一些具体的应用层协议来实现的。目前，提供电子邮件发送服务的协议主要是SMTP（Simple Mail Transfer Protocol，简单邮件传输协议），而提供邮件接收服务的协议主要是POP3（Post Office Protocol 3，邮局协议3）。它们都是工作于C/S模式的，都有对应的客户端和服务端两部分。在“用户代理”部分会同时提供SMTP客户端和POP3客户端服务，而在“消息传输代理”部分则除了包括SMTP服务器和POP3服务器外，还要包括SMTP客户端，因为在发送和接收邮件两方的邮件服务器之间还需要进行电子邮件传输。

说明 在此，仅以SMTP和POP3这两种应用最广的电子邮件协议为例进行介绍。除此之外，提供电子邮件发送服务的协议还有MIME（Multipurpose Internet Mail Extensions，多用途互联网邮件扩展），提

供邮件接收服务的协议还有IMAP（Internet Mail Access Protocol，互联网邮件访问协议）。具体内容将在本章后面介绍。

图11-25显示了一个完整的SMTP/POP3电子邮件传输系统，包括发件方和收件方各自的电子邮件系统及组成。在发件人和收件人计算机中会安装各种用于电子邮件收发的电子邮件客户端软件，也就是“用户代理”，如outlook、foxmail等。在这类软件中会全面包括SMTP客户端和POP3客户端程序，因为一般用户不可能通过此程序只发送，或者只接收电子邮件。

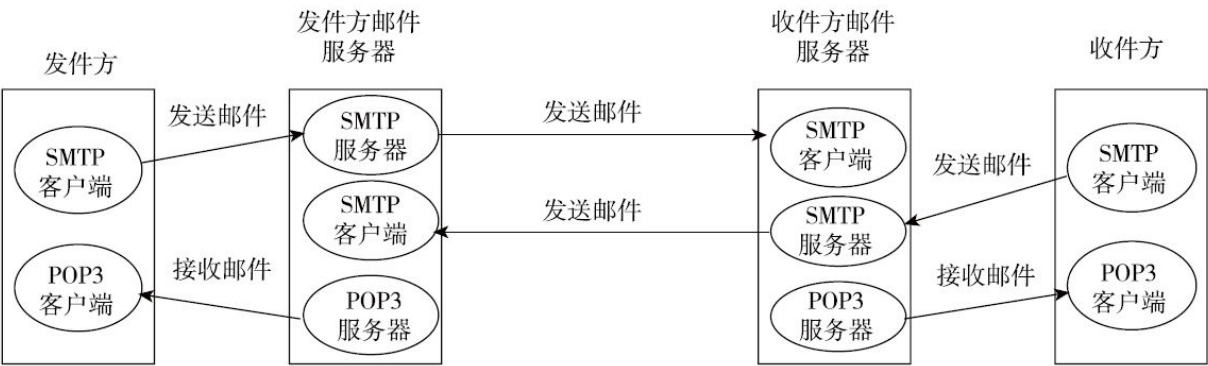


图 11-25 SMTP/POP3电子邮件传输系统

在发件方和收件方邮件服务器中会安装各种电子邮件服务器软件，也就是“消息传输代理”部分，如Exchange Server、SendMial、WinMail、CCMail等。在电子邮件服务器软件中会同时提供SMTP服务器、POP3服务器和SMTP客户端，因为一个电子邮件服务器既要把由用户主动发送过来的电子邮件转发到接收方的电子邮件服务器中（需

要同时利用**SMTP**客户端和**SMTP**服务器），又要允许用户从电子邮件服务器上取回自己的电子邮件（利用的是**POP3**客户端）。

SMTP和**POP3**服务都是使用**TCP**传输层协议进行邮件传输的，各自使用的端口分别为**TCP 25**和**TCP 110**。在如图11-25所示的电子邮件系统中，电子邮件的发送和接收都是通过**TCP**进行的，具体的工作原理将在本章后面介绍。

11.5.2 电子邮件消息格式

要书写和阅读电子邮件，必须先规定标准的电子邮件消息格式。电子邮件消息格式最开始是在RFC822中规定的，这是纯ASCII媒体类型的，后来有改进版的RFC2822，以及扩展的MIME（多用户互联网邮件扩展）格式。本节主要介绍RFC2822电子邮件消息格式和MIME电子邮件消息格式。

1.RFC2822电子邮件消息格式

在RFC2822中规定，一封电子邮件分成两个基本部分：一个信封（也称“邮件头部”）、邮件正文，如图11-26所示。其实它就是一个表单。

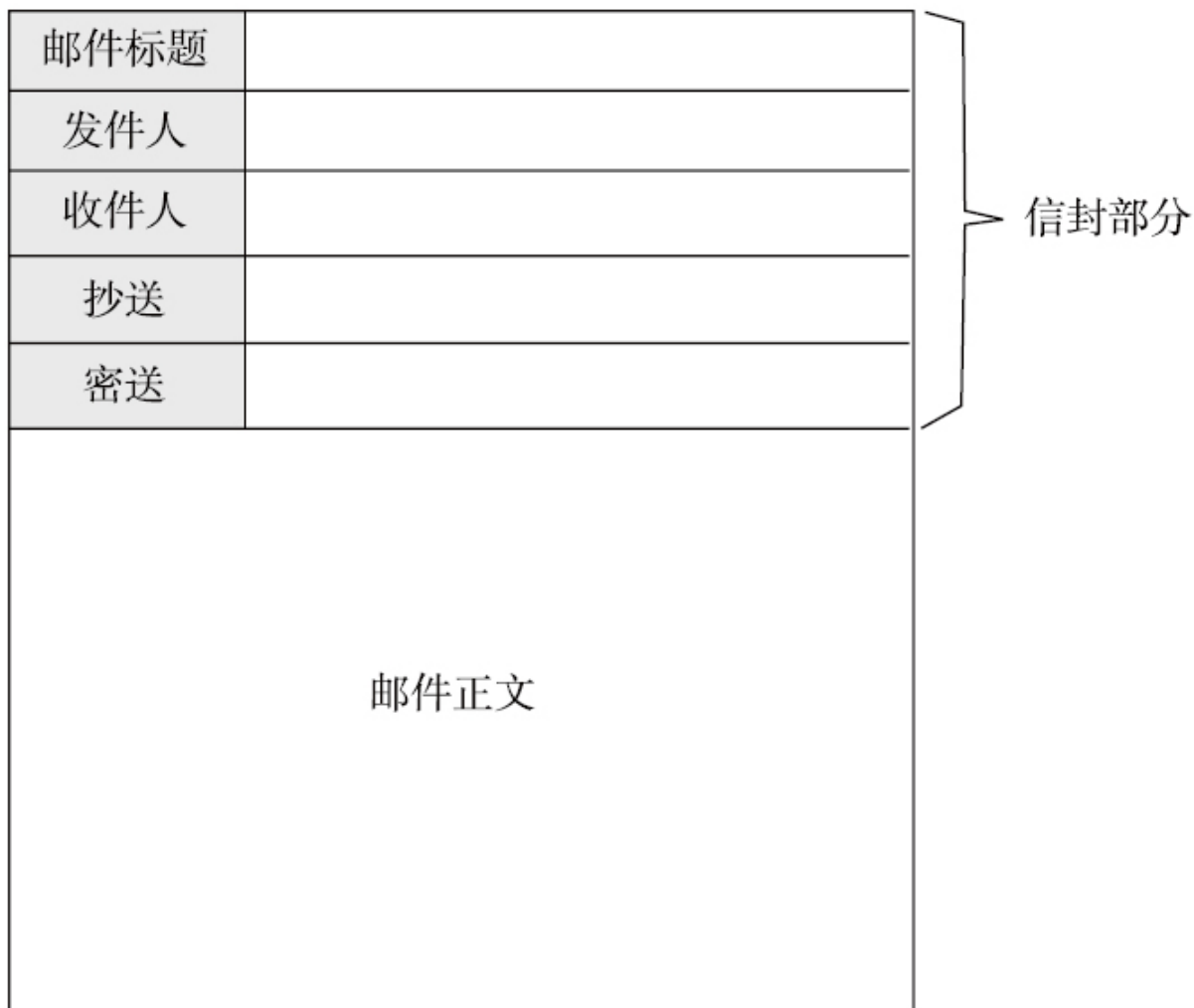


图 11-26 RFC2822电子邮件消息格式

在“信封”部分包括多个消息头，每个消息头占一行，每行包括一个关键字（**keyword**）和一个对应的值（**value**），中间用冒号（:）分隔。常用的关键字包括：**Date**（邮件发送日期）、**From**（发件人）、**Subject**（邮件标题）、**Sender**（写信人）、**To**（收件人）、**Cc**（抄送）、**Bcc**（密送）、**Reply-To**（回信地址）等。在**Subject**关键字后，每次回复都在前面加上一个“**Re**”，从中可以看出这是第几次回复；

Reply-To关键字后面跟的邮箱地址可以与**From**关键字后面跟的发信人邮箱地址相同，也可以不同，因为有的用户有不同的发送邮件地址和接收邮件地址；在“信封”部分中的每个关键字和所对应的值独占一行，在结尾都有一个回车符和换行符。

在“邮件正文”部分是具体的邮件内容，但RFC2822仅支持ASCII文本字符。

2.MIME电子邮件消息格式

前面介绍的RFC2822电子邮件消息格式只能传输ASCII码格式的数据，一些非英语字符消息，像二进制文件、图像、声音等非文字消息都不能在电子邮件中传输。这显然不能满足当前多媒体时代的电子邮件通信需求，于是开发了扩展的电子邮件消息格式标准——RFC1341，然后在RFC2045~RFC2049多个RFC文档中做了修改，这就是MIME。

MIME消息可以包含ASCII文本、图像、音频、视频以及其他应用程序专用的数据，大大扩展了电子邮件消息的数据类型。当然，目前MIME消息格式不仅应用于电子邮件中，还在各种浏览器中普遍应用。每个MIME类型由两部分组成，前面是数据的大类别，如audio（音频）、image（图像）、video（视频）等，后面定义具体的子类。常见的MIME类型如表11-10所示。

表 11-10 常见的 MIME 数据类型

MIEM 类别	MIME 大类 /MIME 子类	扩展名
xml 文档	text/xml	.xml
XHTML 文档	application/xhtml+xml	.xhtml
普通文本	text/plain	.txt
超文本标记语言文本	text/html	.html
RTF 文本	application/rtf	.rtf
PDF 文档	application/pdf	.pdf
Word 文档	application/msword	.doc
Excel 文档	application/msexcel	.xls
PNG 图像	image/png	.png
GIF 图形	image/gif	.gif
JPEG 图形	image/jpeg	.jpeg, .jpg
au 声音文件	audio/basic	.au
MIDI 音乐文件	audio/midi, audio/x-midi	.mid, .midi
RealAudio 音乐文件	audio/x-pn-realaudio	.ra, .ram
MPEG 文件	video/mpeg	.mpg, .mpeg
AVI 文件	video/x-msvideo	.avi
GZIP 文件	application/x-gzip	.gz
TAR 文件	application/x-tar	.tar

在电子邮件格式方面，MIME 也继承了 RFC2822 的格式，分为“信封”和“邮件正文”两部分。在“信封”部分除了仍支持 RFC2822 中的消息头外，另外新增了如表 11-11 所示的消息头。

表 11-11 新增的 MIME 消息头

新增的 MIME 消息头	说 明
MIME-Version	标识使用的 MIME 协议版本，目前都是 1.0 版本
Content-Type	指示消息中的数据类型，以便数据能被适当的处理。参见表 11-10
Content-Transfer-Encoding	指示消息中数据采用的编码格式。可以使用 7bit（7 位元的 ASCII 编码方式）、8bit（8 位元的 ASCII 编码方式）、binary（二进制编码方式）、quoted-printable 和 base64 中的一种编码方式
Content-ID	用于为“multipart/related”组合消息中的内嵌资源指定一个唯一标识号，在 HTML 格式的正文中可以使用这个唯一标识号来引用 该内嵌资源
Content-Description	指示邮件阅读程序处理数据内容的方式，有 inline 和 attachment 两种标准方式，inline 表示直接处理，而 attachment 表示当做附件处理。如果将 Content-Disposition 设置为 attachment，在其后还可以指定 filename 属性
Content-Location	用于为内嵌资源设置一个 URI 地址，这个 URI 地址可以是绝对或相对的。当使用 Content- Location 头字段为一个内嵌资源指定一个 URI 地址后，在 HTML 格式的正文中也可以使用这个 URI 来引用该内嵌资源
Content-Base	用于为内嵌资源设置一个基准路径，只有这样，Content-Location 头字段中设置的 URI 才可以采用相对地址

11.5.3 SMTP请求命令和应答消息

SMTP（对应RFC2821）是目前应用最广泛的提供电子邮件发送服务的应用层协议，使用**TCP**端口号为25，但它是仅可以进行**ASCII**字符传输的协议（注意，这里并不是说电子邮件内容仅可以是**ASCII**字符，而是指**SMTP**通信时采用**ASCII**消息格式）。

前面提到过，**SMTP**也是基于C/S模式的，它也有对应的客户端和服务端。但要注意的是，虽然**SMTP**服务器功能仅是由各种电子邮件服务器软件提供，但**SMTP**客户端功能会可同时在电子邮件客户端软件和电子邮件服务器软件中提供（参见图11-25），因为在整个电子邮件发送过程中，发件方需要利用**SMTP**客户端向自己的电子邮件服务器发送邮件，在发件方的电子邮件服务器把邮件传送到收件方的电子邮件服务器的过程中也要利用**SMTP**客户端。

在发送电子邮件时，源主机的**SMTP**客户端程序要与目的主机的**SMTP**服务器程序的**TCP 25**号端口建立**TCP**传输连接。**SMTP**客户端是以一系列4个字母的**ASCII**格式请求命令向**SMTP**服务器发送请求消息的，而**SMTP**服务器则是以一些非常简单的3位数字格式的**ASCII**消息代码进行应答的，不同的消息代码代表不同的含义。

SMTP邮件发送过程是在基于TCP 25号端口传输连接建立之后进行的，整个邮件传输过程中SMTP客户端和SMTP服务器之间的会话也可以分为连接建立、邮件传输和连接释放3个阶段。在这3个阶段中，每个阶段SMTP客户端使用的请求命令如表11-12所示（目前可用的是11条命令）。每个命令后面都跟一个CRLF（回车和换行标识符）。SMTP服务器的应答消息如表11-13所示。SMTP服务器针对SMTP客户端的各个请求命令可以做出的应答消息如表11-14所示。

表 11-12 SMTP 客户端请求命令

SMTP 会话阶段	请求命令	说 明
连接建立阶段	HELO	后面跟 SMTP 客户端域名，发送方向服务器标识用户身份，请求建立 SMTP 应用会话。命令格式为：HELO（这里要有空格）<域名><CRLF>，如 HELO lychb.com
	EHLO	与 HELO 功能一样，但支持 SMTP 服务扩展的客户端应该以 EHLO 命令开始 SMTP 会话，而不是通常的 HELO 命令。命令格式为：EHLO（这里要有空格）<域名><CRLF>，如 EHLO lychb.com
邮件传输阶段	MAIL FROM	后面跟发件人的电子邮箱地址，向对方服务器标识发件人此次所使用的邮箱地址。命令格式为：MAIL FROM:<reverse-path><CRLF>，其中“reverse-path”包括发件人的电子邮箱地址，如 MAIL FROM:<winda@lycb.com>
	RCPT TO	后面跟接收此电子邮件的邮箱地址，向对方服务器标识单个收件人电子邮箱地址。在 MAIL From 命令后面可有多条 RCPT TO 命令。命令格式为：RCPT TO：其中 <forward-path><CRLF>，其中 forward-path 为收件人的电子邮箱地址，如 grfw@vd.bar.org
	DATA	后面无参数，标识进行邮件传输初始化，准备开始正式的邮件传输过程，在内容的最后以 CRLF 结尾，标识邮件传输结束。命令格式为：DATA<CRLF>，如 DATA
	VRFY	后面跟要确认的用户 / 电子邮箱，请求服务器验证指定的用户账户或电子邮箱是否存在，考虑到用户电子邮箱账户安全性因素，服务器一般禁止此操作。命令格式为：VRFY（这里要有空格）<校验的字符串><CRLF>，如 VRFY winda@lycb.com
	EXPN	后面跟要确认的电子邮箱列表，请求服务器验证指定的电子邮箱列表是否存在，同样考虑到用户电子邮箱账户安全性因素，服务器一般禁止此操作。命令格式为：EXPN（这里要有空格）<校验的字符串><CRLF>，如 EXPN Example-People
	HELP	后面跟要请求帮助的命令，请求服务器提供帮助信息。命令格式为：HELP（这里要有空格）<要查询帮助信息的命令><CRLF>，HELP NOOP（就是想向服务器查询 NOOP 命令的帮助信息）
	NOOP	后面无参数，无操作，要求服务器应答 OK，用于测试。命令格式为：NOOP<CRLF>，如 NOOP
	RSET	后面无参数，重置 SMTP 会话，当前传输被取消。命令格式为：RSET<CRLF>，如 RSET
连接释放阶段	QUIT	后面无参数，终止邮件会话，要求服务器应答 OK。命令格式为：QUIT<CRLF>，如 QUIT

表 11-13 SMTP 服务器应答消息

消息代码	说 明
200	非标准的成功应答, 参见 RFC876
211	显示服务器系统状态或系统帮助信息
214	显示帮助消息
220	显示的域 SMTP 服务已准备就绪
221	显示的域 SMTP 服务关闭了传输信道
250	操作正确完成, OK
251	用户非本地, 将转发到显示的域中
354	开始邮件输入, 以 <CRLF>.<CRLF> 结束, CRLF 为回车和换行标识符
421	指定域的 SMTP 服务不可用, 关闭传输信道
450	请求的邮件操作未完成, 邮箱不可用
451	服务器本地处理进程出错, 放弃请求的邮件操作
452	服务器系统存储空间不足, 请求的操作未执行
500	语法错误, 命令不可识别
501	命令参数或格式语法错误
502	命令不可执行
503	错误的命令序列
504	命令参数未执行
521	显示的域不接收邮件
530	拒绝访问
550	邮箱不可用, 请求的邮件操作不执行
551	非服务器本地用户, 可尝试所显示的域
552	超出了对应用户的邮件存储分配空间, 放弃请求的邮件操作
553	用户邮箱名不允许, 不执行请求的邮件操作
554	邮件传输失败

表 11-14 各 SMTP 请求命令可能返回的应答消息

SMTP 客户端请求命令	可能返回的消息代码	说 明
HELO	250	请求的邮件操作正确完成, OK
	500	语法错误, 命令不可识别
	501	命令参数语法错误
	504	命令参数未执行
	521	显示的域没有接收邮件
	421	显示域的 SMTP 服务不可用, 关闭传输信道
EHLO	250	请求的邮件操作正确完成, OK
	550	不执行
	500	语法错误, 命令不可识别

(续)

SMTP 客户端请求命令	可能返回的消息代码	说 明
EHLO	501	命令参数语法错误
	504	命令参数未执行
	421	显示域的 SMTP 服务不可用, 关闭传输信道
MAIL	250	请求的邮件操作正确完成, OK
	552	超出用户邮件存储空间限制, 请求的邮件操作放弃
	451	服务器本地进程错误, 请求的邮件操作放弃
	452	服务器系统存储空间不足, 请求的邮件操作未发生
	500	语法错误, 命令不可识别
	501	命令参数语法错误
	421	显示域的 SMTP 服务不可用, 关闭传输信道
RCPT	250	请求的邮件操作正确完成, OK
	251	非服务器上本地用户, 将转发到显示的邮箱地址
	550	邮箱不可用, 请求的邮件操作未发生
	551	非服务器上本地用户, 将尝试显示的邮箱地址
	552	超出用户存储空间分配限制, 请求的邮件操作放弃
	553	用户邮箱名不允许, 请求的邮件操作未发生
	450	用户邮箱不可用, 请求的邮件操作未发生
	451	本地进程错误, 请求的邮件操作放弃
	452	服务器系统存储空间不足, 请求的邮件操作未发生
	500	语法错误, 命令不可识别
	501	命令参数语法错误
	503	错误的命令序列
	521	显示的邮件域不接收邮件
	421	显示域的 SMTP 服务不可用, 关闭传输信道
	354	开始邮件传输
DATA	451	本地进程错误, 请求的邮件操作放弃
	554	传输失败
	500	语法错误, 命令不可识别
	501	命令参数语法错误
	503	错误的命令序列
	421	显示域的 SMTP 服务不可用, 关闭邮件接收传输信道
	250	请求的邮件操作正确完成, OK
	552	超出用户存储空间分配限制, 请求的邮件操作放弃
	452	服务器系统存储空间不足, 请求的邮件操作未发生
RSET	200	非标准成功应答
	250	请求的邮件操作正确完成, OK
	500	语法错误, 命令不可识别

(续)

SMTP 客户端请求命令	可能返回的消息代码	说 明
RSET	501	命令参数语法错误
	504	命令参数未执行
	421	显示的域 SMTP 服务不可用, 关闭邮件接收传输信道
VRFY	250	请求的邮件操作正确完成, OK
	251	非服务器上本地用户, 将转发到显示的邮箱地址
	550	邮箱不可用, 请求的邮件操作未发生
	551	非服务器上本地用户, 将尝试显示的邮箱地址
	553	用户邮箱名不允许, 请求的邮件操作未发生
	500	语法错误, 命令不可识别
	501	命令参数语法错误
	502	命令未执行
	504	命令参数未执行
	421	显示域的 SMTP 服务不可用, 关闭邮件接收传输信道
	250	请求的邮件操作正确完成, OK
	550	邮箱不可用, 请求的邮件操作未发生
EXPN	500	语法错误, 命令不可识别
	501	命令参数语法错误
	502	命令未执行
	504	命令参数未执行
	421	显示域的 SMTP 服务不可用, 关闭邮件接收传输信道
	211	系统状态或系统帮助应答
HELP	214	帮助消息
	500	语法错误, 命令不可识别
	501	命令参数语法错误
	502	命令未执行
	504	命令参数未执行
	421	显示域的 SMTP 服务不可用, 关闭邮件接收传输信道
NOOP	200	非标准成功应答
	250	请求的邮件操作正确完成, OK
	500	语法错误, 命令不可识别
	421	显示域的 SMTP 服务不可用, 关闭邮件接收传输信道
QUIT	221	显示域的 SMTP 服务关闭传输信道
	500	语法错误, 命令不可识别

说明 在RFC822中, SMTP还有几条已废弃的命令, 其中SEND、SOML和SAML现在都直接用MAIL FROM命令替代, TURN被QUIT命令所取代。

11.5.4 SMTP服务的工作原理

SMTP服务工作在以下两种情况：一是从在发件方把电子邮件传输到发件方电子邮件服务器上；二是从发件方电子邮件服务器传输到收件方电子邮件服务器上。它们都是通过两端直接建立的TCP传输连接进行电子邮件传输的，接收电子邮件的SMTP服务器端使用TCP 25号端口。

1.SMTP邮件传输的3个阶段

SMTP客户端使用“推”（push）的方式发送电子邮件，每个电子邮件传输的SMTP服务应用过程均包括：连接建立（这是指SMTP应用会话连接建立阶段，不是TCP传输连接建立阶段）、邮件传输和连接释放3个阶段（注意，这3个阶段都是在TCP传输连接建立后进行的）。有关这3个阶段所用的请求命令的应答消息可参见11.5.3节的介绍。下面介绍这3个阶段的基本流程，其中，“发件方”既可以是发件方主机，也可以是发件方电子邮件服务器；“收件方”既可以是发件方电子邮件服务器，也可以是收件方电子邮件服务器，具体要视所进行的电子邮件传输过程是从发件方主机发送到发件方电子邮件服务器，还是从发件方电子邮件服务器到收件方电子邮件服务器而定。

（1）连接建立阶段

在SMTP应用会话连接建立阶段，当发件方有电子邮件发送时，首先安装在其主机中的SMTP客户端程序会与所配置的电子邮件服务器建立好基于TCP 25号端口的传输连接。以后所有的电子邮件传输过程都是在已建立好的TCP连接基础上进行的。

TCP传输连接建立好后，发件方调用SMTP客户端服务发送HELO命令，后面跟发件方的DNS域名，向收件方表明自己的身份。如果能通过接收方的身份验证，那么收件方会返回一条类似于“250 OK”的应答消息，表示收件方已接受会话连接建立请求，并建立好了会话连接；如果在发送连接请求的HELO命令中出现了差错，或者收件方当前SMTP服务不可用，则会显示对应的错误代码，具体参见表11-14。

（2）邮件传输阶段

发件方在发送完HELO命令，并且收到收件方返回的250消息代码后，即表明收件方已准备就绪，进入到了邮件传输阶段，发件方可以立即通过SMTP客户端发送邮件。在邮件传输阶段，SMTP客户端可以使用多个命令，如用MAIL FROM命令指示发件人的邮箱地址，用RCPT TO命令指示收件人的邮箱地址，且可以发送多条RCPT TO命令，每条命令包括一个收件人邮箱地址。

当以上两条命令发送后，发件方如果都收到了收件方返回的类似于“250 OK”的应答消息，即表示收件方已正确执行了该命令，此时发

件方SMTP客户端就可以通过DATA命令进行正式的邮件内容传输了。在邮件内容全部传输完后，还要传输一条<CRLF>.<CRLF>消息，以标识邮件内容的结束。收件方在收到最后一条消息后会返回类似于“250 OK”的应答消息代码。

如果有另外一封邮件要传输，那么它会继续重复上述的邮件传输过程，但无须重复连接建立阶段。

(3) 连接释放阶段

当没有邮件传输时，发件方SMTP客户端要发送一条QUIT命令结束本次SMTP应用会话进程。正常情况下，收件方会返回一条代码为221的应答消息，表示服务器已接受关闭连接请求，释放本次SMTP应用会话连接。

2.SMTP邮件传输流程示例

为了理解上述邮件传输流程，下面仅以发件方电子邮件服务器把邮件传送到收件方电子邮件服务器的过程为例介绍SMTP邮件传输过程中的3个阶段。假设发件方电子邮件服务器的域名为lycb.com，发件人的电子邮箱地址为winda@lycb.com，收件方的电子邮件服务器域名为grfw.com。基本过程如图11-27所示（仅考虑基本流程和正常情况下的应答代码，不考虑各种错误）。

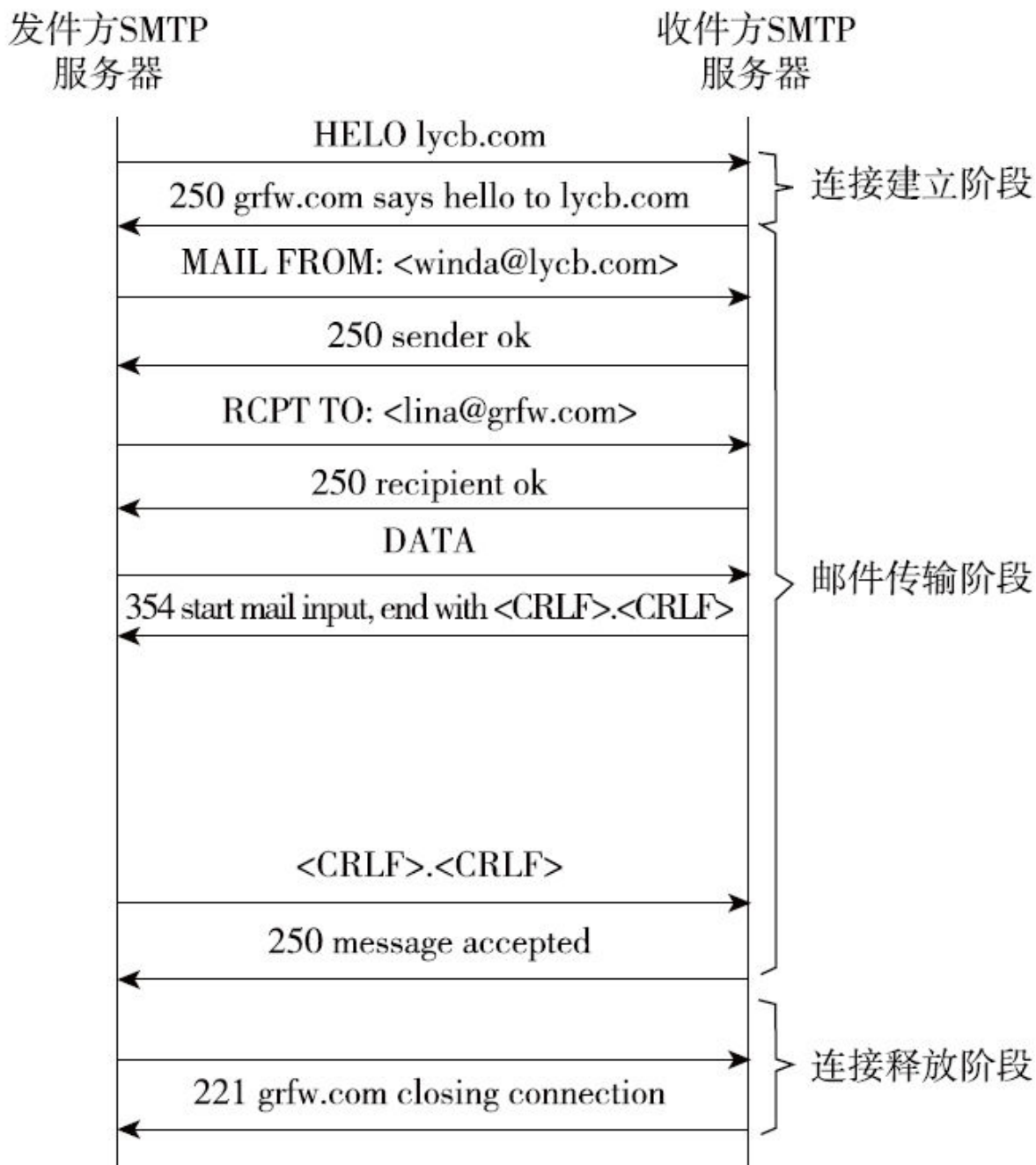


图 11-27 SMTP邮件传输3个阶段的基本请求命令和应答流程

下面是一个典型的SMTP邮件传输示例的脚本，其中C表示SMTP客户端发送的请求命令，S代表SMTP服务器返回的应答消息。其中各

请求命令和应答消息代码含义参见11.5.3节相关内容。

S:220 foo.com Simple Mail Transfer Service Ready !--- 首先, SMTP服务器监听到TCP25号端口上有连接请求, 建立好相应的TCP连接并通知SMTP客户端, 此时说明它已做好了准备, 可以正式进行邮件传输

C:<open connection> !--- SMTP客户端打开已建立好的TCP连接

C:HELO bar.com

S:250-foo.com greets bar.com

S:250-8BITMIME

S:250-SIZE

S:250-DSN

S:250 HELP

C:MAIL FROM:<Smith@bar.com>

S:250 OK

C:RCPT TO:<Jones@foo.com>

S:250 OK

C:RCPT TO:<Green@foo.com>

S:550 No such user here

C:RCPT TO:<Brown@foo.com>

S:250 OK

C:DATA

S:354 Start mail input;end with<CRLF>.<CRLF>

C:Blah blah blah...

C:...etc.etc.etc.

C:.

S:250 OK

C:QUIT

S:221 foo.com Service closing transmission channel

11.5.5 POP3请求命令及应答消息

上面介绍的SMTP服务采用“推”的方式可以把邮件发送到发件方自己的电子邮件服务器，然后由发件方的电子邮件服务器转发到收件方的电子邮件服务器上。但是，收件方的电子邮件服务器不会通过SMTP服务再把邮件推送到收件人主机上，因为收件人并不知道什么时候有人发邮件给他，也不可能24小时在线，即不可能随时接收。我们希望的是像发件人发送邮件那样，在需要的时候随时进行接收。这时就得靠POP3这样具有“拉”（pull）功能的邮件服务了。

POP3（对应RFC1939）是POP（对应RFC918）的第3个版本，解决了POP服务器在用户阅读了某封邮件后即删除该邮件的不足，POP3服务器允许用户把邮件存储在邮件服务器上。POP3允许用户从服务器上把邮件存储到本地主机（即自己的计算机）上，同时根据客户端的需要删除或保存在电子邮件服务器上的邮件。如果把SMTP服务器称为“发送邮件服务器”（用来发送邮件），则POP3服务器就可以称为“接收邮件服务器”（用来接收邮件）。

POP3也是工作在C/S模式上的，其中POP3客户端位于安装了各种电子邮件客户端软件的用户主机上，而POP3服务器则位于安装了各种电子邮件服务器软件的服务器主机上，参见图11-25所示的电子邮件系统结构。POP3客户端和POP3服务器之间的通信是在TCP 110号端口所

建立的TCP传输连接基础之上的。POP3服务器会时刻在TCP 110端口上进行监听，发现有POP3客户连接请求即做出相应的应答。但POP3服务仅用于用户从自己的电子邮件服务器上获取电子邮件，不像SMTP服务那样涉及两个过程，相对来说较为简单。

1.POP3请求命令

与SMTP服务一样，POP3应用服务通信也是先由POP3客户端发出请求命令，然后POP3服务器返回应答消息。在通过POP3服务器的身份验证后，POP3客户端才可进行邮件接收或者从POP3服务器删除邮件等操作。但POP3同样是只支持ASCII的通信交互方式，因此，无论是POP3客户端发出的请求命令，还是POP3服务器向POP3客户端返回的应答消息，都是ASCII码格式。

POP3客户端发出的请求命令是3~4个字母的ASCII码格式。每个POP3请求命令由一个命令关键字和一些参数组成，各部分之间以空格分隔，每个参数长度可达40个字符，所有命令均以一个<CRLF>标志符结束。具体命令说明如表11-15所示（共12条）。

表 11-15 POP3 客户端请求命令

USER name	指示用户访问 POP3 邮件服务器时所用的用户邮箱账户名（不包括域名部分），它与下面 PASS 命令指出的账户和密码一起作为访问 POP3 服务器时进行身份验证的凭据。如 USER winda
PASS string	指示用户访问 POP3 邮件服务器时所用的用户邮箱账户和密码，它与上面的 USER 命令指出的账户和密码一起作为访问 POP3 服务器时进行身份验证的凭据
APOP name digest	指示用户访问 POP3 服务器所使用的加密密钥（避免以明文方式在网上传输密码），name 参数是用户邮箱账户名，digest 为通过 MD5 协议加密后的密钥
STAT	查询用户自己的邮箱统计信息。最开始显示的是“OK”，然后空一格后再显示邮件的总数，再空一格显示所有邮件的大小等内容
LIST [msg]	请求查看指定（选择 msg 参数时）或者全部（不选择 msg 参数时）邮件汇总信息。服务器应答后会显示每个列出的邮件的邮件号和对应的邮件大小
RETR msg	请求接收指定邮件
DELE msg	请求给指定邮件做删除标记。做了删除标记的邮件不会真正被删除，因为只有在 POP3 客户端发出 QUIT 命令后才会真正删除这些做了删除标记的邮件
NOOP	无操作，用于检测 POP3 客户端与 POP3 服务器的连接情况，仅用于测试
RSET	恢复所有邮件均为无删除标记，也就是清除所有通过 DELE 命令所做的邮件删除标记
TOP msg lines	请求查看指定邮件（通过 msg 参数指定）中指定行（通过 lines 参数指定）前的内容
UIDL [msg]	请求查看指定（选择 msg 参数时）或所有（不选择 msg 参数时）邮件的内容，但不能查看有删除标记的邮件
QUIT	用户邮件接收完成，请求结束本次 POP3 应用会话。如果此时某邮件消息打上了删除标记，则服务器会删除该邮件（不在服务器上保留备份）；如果在删除邮件时发生了错误，则可能会有一些或者全部已标记为删除的邮件均不会被删除。无论删除是否成功，服务器均会释放对应的 POP3 应用会话连接，不允许进行访问，同时关闭对应的 TCP 连接

2.POP3应答消息

POP3服务器对POP3客户端请求命令的应答消息是由一个状态码和一些附加信息组成的。应答消息可以占多行，但每行的最后必须加<CRLF>，而整个消息的结束要以<CRLF>.<CRLF>表示。

大多数POP3请求命令（有些命令只可能返回一种状态码）都可能返回两种不同的应答状态码：“+OK”（确定）和“-ERR”（错误）。注意，“OK”和“ERR”必须大写。表11-15中的POP3客户端请求命令可能返回的应答消息如表11-16所示（注意，其中返回的应答消息仅为示例）。

表 11-16 POP3 服务器应答消息

POP3 客户端 请求命令	POP3 服务器应答消息示例	说明
USER	+OK mrose is a valid mailbox	正确，该用户邮箱可用
	-ERR sorry, no mailbox for frated here	错误，没有指定的邮箱用户
PASS	+OK mrose's maildrop has 2 messages (320 octets)	正确，并返回对应用户邮箱的基本统计信息，包括邮件数和邮件总体大小
	-ERR maildrop already locked	错误，指定用户邮箱已被锁定
APOP	+OK maildrop has 1 message (369 octets)	正确，并返回对应用户邮箱的基本统计信息，包括邮件数和邮件总体大小
	-ERR permission denied	错误，拒绝访问
STAT	+OK 2 320	确认，返回对应用户邮箱邮件数和邮件总体大小
LIST	+OK 2 messages (320 octets)	正确，返回对应邮件的邮件号和大小，并开始接收邮件
	-ERR no such message, only 2 messages in maildrop	错误，没有对应邮件号的邮件
RETR	+OK 120 octets	正确，返回对应邮件大小
	-ERR no such message	错误，没有对应邮件号的邮件
DETE	+OK message 1 deleted	正确，指定邮件已标记为删除
	-ERR message 2 already deleted	错误，指定邮件已标记为删除
NOOP	+OK	确认，无返回
REST	+OK maildrop has 2 messages (320 octets)	确认，返回已恢复删除标记的邮件数和邮件总体大小
TOP	+OK	正确，返回指定行的邮件内容
	-ERR no such message	错误，没有对应邮件内容
UIDL	+OK 2 QhdPYR:00WBw1Ph7x7	正确，返回对应邮件号的唯一 ID 号
	-ERR no such message, only 2 messages in maildrop	错误，没有对应邮件号的邮件
QUIT	+OK dewey POP3 server signing off	确认，对应服务器正在释放 POP3 应用会话连接

11.5.6 POP3服务的工作原理

从理论上讲，也可以把整个POP3邮件接收过程分为“连接建立”、“邮件接收”和“连接释放”这3个阶段。它们都是在基于与POP3服务器TCP 110端口建立TCP传输连接的基础上进行的，也就是在进行正式的邮件接收前，POP3客户端必须与POP3服务器的TCP 110端口建立传输连接（POP3服务器会时刻监听TCP 110端口，一旦有连接请求马上做出应答）。

连接建立阶段其实就是POP3客户端向POP3服务器发出身份验证请求，给出用户名和密码的阶段，包括表11-15所示的USER、PASS和APOP这几个请求命令。

邮件接收阶段是用户在进行邮件接收时进行的各种操作，包括表11-15中除USER、PASS、APOP和QUIT这4个命令外的其他所有命令。

连接释放阶段是POP3客户端在接收完全部邮件，或者进行完所有需要的操作后释放本次POP3应用会话连接的过程，包括的请求命令是QUIT。

图11-28是一个典型的POP3邮件接收的基本流程示例，对应的代码脚本如下所示，其中C代表POP3客户端发送的请求命令，S代表POP3

服务器返回的应答消息。其中各请求命令和应答消息代码含义参见11.5.5节相关内容。

```
S:+OK POP3 server ready<1896.697170952@dbc.mtview.ca.us> !--- 首先, POP3服务器监听到TCP 110号端口上有连接请求, 建立好相应的TCP连接并通知POP3客户端, 此时, 说明它已做好了准备, 可以正式进行邮件接收
C:<open connection> !--- POP3客户端打开已建立好的TCP连接
C:USER winda
S:+OK winda is a real hoopy frood
C:PASS 123456
S:+OK winda's maildrop has 2 messages(320 octets)
C:STAT
S:+OK 2 320
C:LIST
S:+OK 2 messages(320 octets)
S:1 120
S:2 200
S:.
C:RETR 1
S:+OK 120 octets
S:<the POP3 server sends message 1>
S:.
C:DELE 1
S:+OK message 1 deleted
C:RETR 2
S:+OK 200 octets
S:<the POP3 server sends message 2>
S:.
C:DELE 2
S:+OK message 2 deleted
C:QUIT
S:+OK dewey POP3 server signing off(maildrop empty) !--- 邮件接收完成, 邮件投递箱为空
```

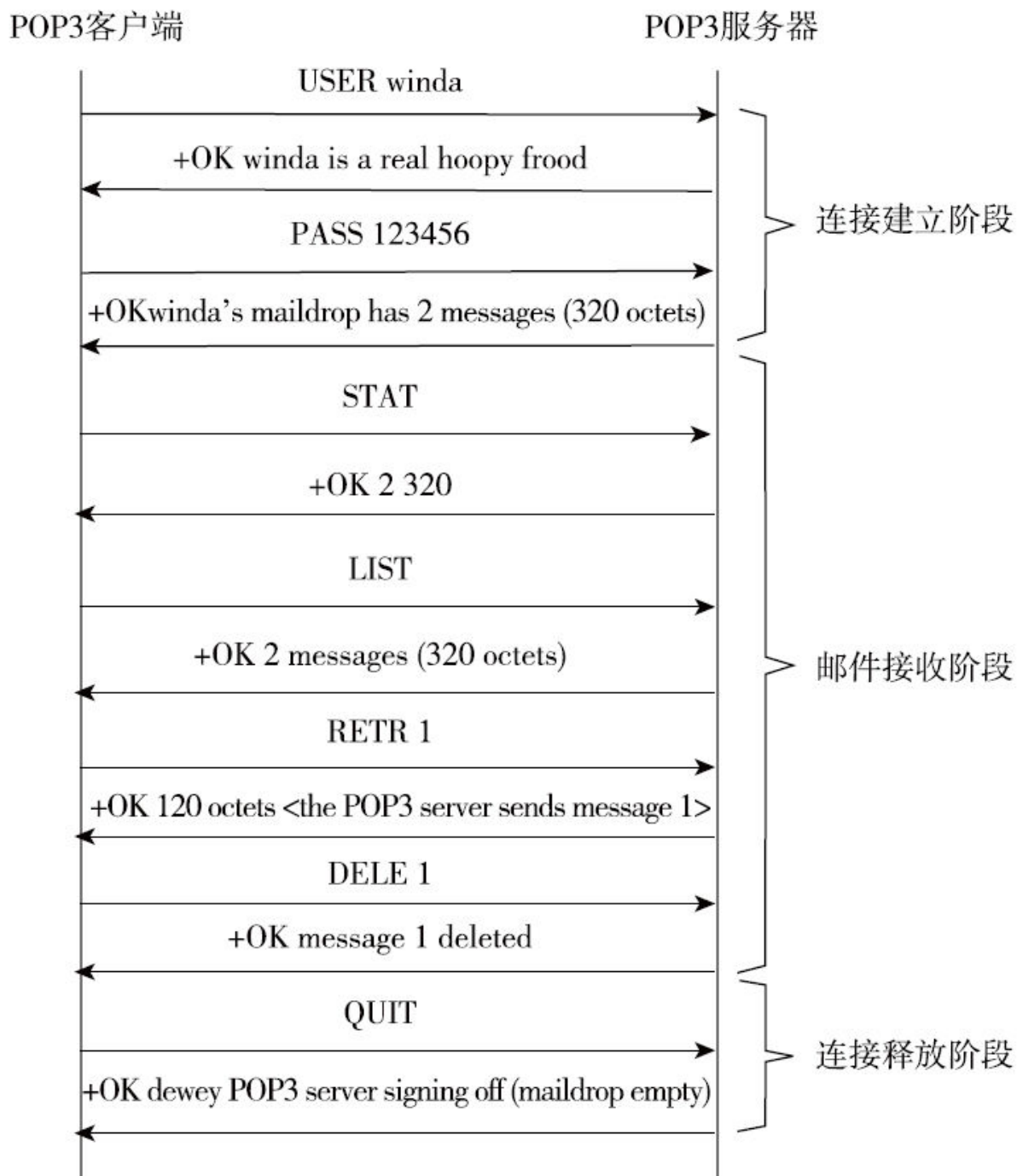


图 11-28 POP3邮件接收基本流程示例

11.5.7 IMAP4简介

在接收邮件方面，除了前面介绍的POP3外，目前还有一种功能更强大，但也更复杂的协议——IMAP4（Internet Message Access Protocol 4，互联网消息访问协议第4版）。IMAP4也是采用C/S工作模式的。IMAP4服务器工作在TCP 143号端口。目前绝大多数电子邮件客户端程序都同时支持POP3和IMAP4，如Outlook、Windows Mail、Outlook Express、Foxmail、Entourage、Mozilla Thunderbird和Eudora等。

与POP3类似，IMAP4（对应RFC3501）也是为用户提供邮件接收服务的（发送邮件同样使用SMTP）。IMAP4改进了POP3的不足，具体来说，它有以下几方面的特性：

（1）支持服务器端邮件副本存储

在默认情况下，POP3客户端程序在将电子邮件下载到计算机上后，会从服务器上删除所有已下载的电子邮件。而IMAP4却可以在电子邮件服务器上保留下载的邮件的副本，这样就可以从多台计算机访问保存在服务器上的同一封电子邮件。

（2）支持离线、在线和断线3种访问方式

POP3仅支持离线访问方式，客户端只会与POP3服务器连接一段时间，直到它下载完所有新信息，客户端即断开连接，而IMAP4可以同时支持离线、在线和断线3种访问方式。

在离线方式中，虽然IMAP4客户端软件也是把邮件存储在本地硬盘上，但IMAP4提供的摘要浏览功能可以让用户通过先阅读完所有的邮件到达时间、主题、发件人、大小等信息后才做出是否下载的决定。也就是说，用户不必等所有的邮件都下载完毕后才知究竟邮件里都有些什么。如果用户根据摘要信息就可以判断某些邮件对他来说毫无用处，那么他可以直接在服务器上把这些邮件删除，而不必浪费宝贵的上网时间。在在线方式中，IMAP4客户端可以通过Web浏览器来访问IMAP4服务器上的邮件，不需要像离线模式那样把邮件下载到本地才能阅读。在断线方式中，IMAP4客户软件把用户选定的消息和附件复制或缓存到本地磁盘上，并把原始副本留存在邮件服务器上，当以后用户重新连接IMAP4服务器时，这些邮件又可以与服务器进行再同步，显示全部的邮件内容。

(3) 支持多个客户同时连接到一个邮箱

POP3在同一时间只能允许一个用户活动连接，而IMAP4允许同一时间不同用户对同一邮箱的多个用户活动连接，并且提供了一种机制能让客户感知当前连接到这个邮箱的其他用户的操作。这对于多个用户使用同一邮箱时非常有用，如一个项目组可能使用同一个邮箱来进

行邮件收发，但不同成员可能会同时上网查看邮箱中各自所需的邮件。

（4）支持选择性获取

如果用户的IMAP客户端软件完整支持IMAP4 rev1的话，则用户还可以享受选择性下载附件的服务，既可以只下载正文，也可以下载部分附件或全部附件。例如，在一封邮件中含有三个附件，但其中只有一个附件是用户所需要的，这时用户可以只下载那一个需要的附件，节省了下载其余两个附件的时间。

（5）支持在用户邮箱上创建、管理多个文件夹功能

在POP3中，每个用户邮箱只能有几个默认的文件夹（如草稿箱、发件箱、收件箱、垃圾箱等），不可新建其他文件夹，也不可删除原来的这些默认文件夹，但在IMAP4服务器上，可以在每个默认文件夹下创建多个新的子文件夹，并且可以对各个邮箱文件夹进行重命名，或进行删除操作。这样，用户就可以对接收或者发送的邮件分门别类地进行管理，就像本地磁盘的文件夹管理一样。

（6）支持服务器搜索

IMAP4提供了一种可以使客户端在服务器上搜索符合多个标准的信息的机制，相当于在本地磁盘中进行文件搜索。IMAP4的搜索功能

很强大，不仅可以基于邮件标题进行搜索，还可基于邮件信封中其他部分，甚至邮件正文内容进行搜索。这样，客户端无须下载邮箱中所有信息就可以完成这些搜索。

(7) 支持客户端和服务端间的拖动操作

一般的IMAP4客户端软件都支持邮件在本地文件夹和服务端文件夹的随意拖动，这样可以让用户轻松地把本地硬盘中的文件存放到服务器上，然后在需要的时候同样方便地取回来，所有的功能仅需要一次鼠标拖放的操作就可以实现，就像在网络中不同主机间的文件操作一样简单。

说明 因为IMAP4相对来说更为复杂，且目前POP3可以满足大多数电子邮件用户的需求，所以在此不对IMAP4进行更加详细的介绍。